

PATENT ABSTRACTS OF JAPAN

(11) Publication number : 05-265781

(43) Date of publication of application : 15.10.1993

(51)Int.Cl.

G06F 9/46

(21) Application number : 04-325628

(71)Applicant : INTERNATL BUSINESS MACH
CORP <IBM>

(22) Date of filing : 04.12.1992

(72)Inventor : DINALLO CHRIS A
KOVAL MICHAEL J
LAWTON WILLIAM W
PAULAT JR MARTIN J
TYLER JOHN G
WINTERS SCOTT L
ALLRAN GARY G

(30)Priority

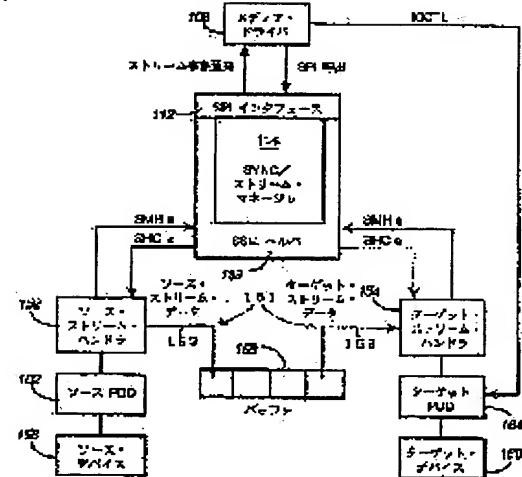
Priority number : 91 816517 Priority date : 31.12.1991 Priority country : US

(54) MULTIMEDIA DATA PROCESSING SYSTEM

(57) Abstract:

PURPOSE: To provide a multimedia system which can transmit a large amount of data continuously in real time.

CONSTITUTION: This system includes a multimedia application program and a multitasking operating system. The extension of the operating system controls data streaming from a source device 158 to a target device 160 to provide real-time continuous streaming. The extension provides central buffer management by a user buffer option, the support of 2-level priority for a data stream handler, the support of an interleaved stream, and the event detection and information of a data stream.



LEGAL STATUS

[Date of request for examination] 04.12.1992

[Date of sending the examiner's decision of rejection] 01.08.1995

[Kind of final disposal of application other than the examiner's decision of rejection or



application converted registration]

[Date of final disposal for application]

[Patent number] 2521016

[Date of registration] 17.05.1996

[Number of appeal against examiner's decision
of rejection] 07-23828

[Date of requesting appeal against examiner's
decision of rejection] 30.10.1995

[Date of extinction of right] 17.05.1999

Copyright (C); 1998,2003 Japan Patent Office



(19)日本国特許庁 (JP)

(12) 特許公報 (B2)

(11)特許番号

第2521016号

(45)発行日 平成8年(1996)7月31日

(24)登録日 平成8年(1996)5月17日

(51)Int.Cl. ⁶ G 0 6 F 13/38	識別記号 3 4 0 3 4 0	序内整理番号 9188-5E	F I G 0 6 F 13/38	技術表示箇所 3 4 0 A 3 4 0 C
--	------------------------	-------------------	-------------------------	------------------------------

請求項の数3(全38頁)

(21)出願番号 特願平4-325628

(22)出願日 平成4年(1992)12月4日

(65)公開番号 特開平5-265781

(43)公開日 平成5年(1993)10月15日

(31)優先権主張番号 8 1 6 5 1 7

(32)優先日 1991年12月31日

(33)優先権主張国 米国(US)

前置審査

(73)特許権者 390009531

インターナショナル・ビジネス・マシンズ・コーポレイション

INTERNATIONAL BUSINESS MACHINES CORPORATION

アメリカ合衆国10504、ニューヨーク州
アーモンク(番地なし)

(72)発明者 ク里斯・エイ・ディナロ

アメリカ合衆国33431、フロリダ州ボ
カ・ラトン、ハーバー・ドライブ 640

(74)代理人 弁理士 坂口 博(外2名)

審査官 吉岡 浩

最終頁に続く

(54)【発明の名称】マルチメディア・データ処理システム

(57)【特許請求の範囲】

【請求項1】プロセッサと、

複数のバッファを有し、少なくとも1つのマルチメディア・アプリケーション・プログラム及びマルチタスキング・オペレーティング・システムを記憶するメモリと、
メディア・データを供給するソース・デバイスと、
前記メディア・データを受け取るターゲット・デバイスと、
前記オペレーティング・システムの下に動作し、前記ソース・デバイスから前記ターゲット・デバイスへデータをストリーム転送するデータ・ストリーミング手段とを備え、

前記データ・ストリーミング手段が、
前記アプリケーション・プログラム中のストリーム生成命令の実行に応答して、前記ソース・デバイスから前記

バッファへデータをストリーム転送するためのソース・
スレッドを生成し、前記アプリケーション・プログラム
中の開始命令に応答して、前記ソース・スレッドを実行
することにより前記ソース・デバイスから前記バッファ
へデータを読み込むソース・ストリーム・ハンドラと、
前記ストリーム生成命令の実行に応答して、前記バッファ
から前記ターゲット・デバイスへデータをストリーム
転送するためのターゲット・スレッドを生成し、前記開始
命令に応答して前記ターゲット・スレッドを実行する
ことにより前記バッファから前記ターゲット・デバイス
へデータを読み出すターゲット・ストリーム・ハンドラ
と、

前記ソース・デバイスからのデータの終わりに達するま
で前記ソース・スレッド及び前記ターゲット・スレッド
を交互に実行させることによって前記ストリーム転送を

実時間で連続的に行うためのストリーム・マネージャとを含み、

前記ソース・スレッド及び前記ターゲット・スレッドは前記オペレーティング・システムの下で選択的に実行されるディスパッチ可能な実行単位であることを特徴とする。

マルチメディア・データ処理システム。

【請求項2】前記プロセッサはオペレーションの異なる優先レベルを有し、各前記ストリーム・ハンドラは、ある優先レベルにおいて実行されるダイナミック・リンク・ライブラリ(DLL)・ストリーム・ハンドラと、前記ある優先レベルよりも高い優先レベルにおいて実行される物理デバイス・ドライバ(PDD)・ストリーム・ハンドラを含むこと、

を特徴とする請求項1記載のマルチメディア・データ処理システム。

【請求項3】前記ソース・デバイスからの前記データは順次的にインタリープされた複数のレコードに含まれ、前記ソース・ハンドラは各バッファを前記レコードにより充填すること、

を特徴とする請求項1又は2記載のマルチメディア・データ処理システム。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明はデータ処理分野に関し、特に、マルチタスキング・オペレーティング・システムの制御の下で、データ・ストリーム転送を連続的に、実時間方式で発生するパーソナル・コンピュータを基本とするマルチメディア・システムに関する。

【0002】

【従来の技術】マルチメディア・システムはテキスト、グラフィックス、ビデオ、イメージ、アニメーション、オーディオなどの種々の組合せにおいて、様々なマルチメディア材料を提供する。こうしたシステムはハードウェアとソフトウェアの組合せにより構成される。ハードウェアには種々のマルチメディア装置が接続されるパーソナル・コンピュータが含まれる。ハードウェアはオペレーティング・システム及びマルチメディア・アプリケーション・プログラムの制御の下で機能する。

【0003】マルチメディア・アプリケーションは装置間、或いはシステム・メモリから装置へ、或いはその反対方向へ、連続的に実時間で大量のデータを転送するために、オペレーティング・システムに重い負荷を課す。マルチメディア・システムはこれらの大きなデータ・オブジェクトを転送するための柔軟でしかも一貫した手段を支持し、またこの活動を実時間で正確に制御しなければならない。新たなマルチメディア装置及びデータ・タイプを追加するためには、最小限度の新たなシステム拡張コードが必要となる。実行時に必要な実メモリのトータルは、システム性能が低下しないように最小化されね

ばならない。また、複雑なデータ・タイプ及びインタリープ式のデータ・オブジェクトを操作する装置の支持も必要である。さらに、システムの最適な特権レベルにおいて、各マルチメディア・データ転送制御手段が実現されなければならない。

【0004】マルチメディア・データ・タイプ及び装置を支持するオペレーティング・システムの拡張は、多数の異なるマルチメディア入出力装置を制御し、また実時間の制限範囲において、大量のマルチメディア・データ・オブジェクトを転送或いはストリームする機能を提供することが必要がある。

【0005】マルチメディア・アプリケーションは、ビットマップの連続的な表示、デジタル化オーディオ波形或いはMIDIオーディオ・シーケンス、或いはアナログ・マイクロフォンまたはライン源からのデジタル化オーディオ入力などの大量のデータの入出力を制御する。アプリケーションはこれら全てのデータ・フローを実時間クロックにより制御する。すなわち、プログラムにより制御される特定の事象は、時間的に明確に定義された時点において発生されなければならない、これらの時点は非常に正確に(例えばミリ秒単位で)定義される。

【0006】OS/2制御プログラム・サービス(DOS呼出、或いはAIXまたはUNIXなどの他のオペレーティング・システムにおける類似のサービス)だけを考えると、アプリケーション・プログラム・インターフェースにおいてこのレベルの機能を制御するには、非常に複雑で装置依存性の高いデータ転送制御モジュールを要する。こうしたモジュールが生成されたとしても、各入出力オペレーションを制御するスレッド(thread)がその要求時間内に実行される保証はない。この問題を指摘すると、アプリケーションは複雑なセマフォー(semaphore)論理を追加し、入出力制御スレッドを時間的に厳密なもの、すなわちタイム・クリティカル・スレッドにする必要が生じる。マルチメディア・アプリケーションでは一般にデータ・スループットが高負荷であるから、マルチタスキング・オペレーティング・システムのもとでは、しばしば、割当時間間隔内にデータを宛先装置に転送することが阻害される。これらの実時間要求を満たせないと、結果的にマルチメディア表現における視覚的或いは聴覚的な欠陥を発生させる。

【0007】

【発明が解決しようとする課題】実時間における連続的な大量のデータの転送要求は、一般化された転送機構が解決しなければならない一連の相互に依存する問題を提示する。これらの問題を以下に示す。

【0008】1. 非常に負荷の大きいデータ入出力要求を有するスループット重視のアプリケーション解像度が640x480x16の非圧縮動画ビデオでは、およそ3Mb/秒を必要とする。例え高度に圧縮されたとしても、デジタル動画ビデオ・データ・ストリ

ームは60Kb／秒を必要とする。8あるいは16ビットのデジタル・オーディオ波形のデータ転送をシナリオに追加すると、スループットの負荷は80Kb／秒あるいはそれ以上に増加する。このスループット負荷を連続的に支持する一方で、マルチタスキング・システムがファイル入出力及びキーボード／マウス装置の制御などの他のアプリケーション機能を実行することが、一般化されたデータ・ストリーミング機構において最も重要な要求である。しかしながら、ユーザ・インターフェース（聴覚的及び視覚的）におけるマルチメディア・データ表示は実時間依存であるため、これらのデータ転送は非常にタイトな実時間制限において発生しなければならない。

【0009】2. 複数の異なるハードウェア入出力装置の制御

マルチメディア・アプリケーションは典型的には種々の入出力装置を制御し、ユーザ・インターフェースのオーディオ及びビデオ分野における表現を管理する。波形獲得及び再生を支援するオーディオ装置、デジタル・オーディオ・コンパクト・ディスク再生、楽器デジタル・インターフェース（MIDI）入出力、及び音声発生などが含まれる。同様に、NTSC（或いはPAL）ビデオのデジタル化、圧縮／伸長、及び獲得を支援するビデオ装置が、ルーチン的に、マルチメディア提示及び／あるいはシナリオに含まれる。一般化されたデータ・ストリーミング機構は、マルチメディア・システムの拡張に関する大きな変更を要求することなく、これらの及びその他のマルチメディア装置を柔軟に支持する必要がある。

【0010】3. 装置及びデータ・タイプ間における一貫した制御サービスの提供

データ・ストリーム機構は、マルチメディアの創作及び表現の過程に含まれる異なる装置及びデータ・タイプ間で、一貫したサービスによって制御されねばならない。マルチメディア・データ標準が発展する過程において新たなデータ・タイプが導入される時、この一般化されたストリーミング機構は実質的に影響されねばならない。制御サービスは新たなデータ・タイプを現存のアプリケーションに容易に組み込むことが必要とされる。同様に、これらのサービスは、現存のアプリケーションに影響を及ぼすことなく、あるいはマルチメディア拡張の大きな変更を必要とせずに、新たな入出力装置のタイプ及び機能を途切れることなく追加する配慮が必要である。

【0011】4. 複数のシステム特権レベルにおける同一制御サービスの提供

マルチメディア・データ・オブジェクトは、ハードウェア入出力装置或いはシステム（またはアプリケーション）・メモリにおいて開始（或いは終了）される。データ・ストリーミングがハードウェア装置と共に開始或いは終了すると、伝送機構はハードウェア装置の振舞いを直接的に制御し、これにより装置依存型となるのではなく、間接的に制御できなければならない。しかしながら

ら、最近のオペレーティング・システムにおいて共通な特徴である、複数の特権レベルを支持する性質においては、最適性能を出すために、伝送機構（“ストリーム・ハンドラ”）が装置固有の物理デバイス・ドライバに最高のシステム特権レベルで接続されることが必要とされる。これはハードウェア装置ドライバが通常実行されるレベルである。このように性能的な理由により、デバイス・ドライバ・レベルのストリーム・ハンドラを作成することが望ましいが、ストリーム・ハンドラ・デバイス・ドライバはそれ自身ハードウェア依存であってはならない。全てのハードウェア特有のコードは、デバイス固有の物理デバイス・ドライバに含まれる必要がある。あるいは、データ・ストリームの出所または宛先として物理的装置が含まれない場合には、ストリーム・ハンドラはより低い特権レベルで実行され、しかも最適性能を達成可能である。このようにデータ伝送機構は複数の特権層に渡り分散されている必要があるため、同一のストリーム制御サービスが、ストリーム・ハンドラが実行される所とのレベルにおいて提供されねばならない。異なる特権レベルにおける複数の入力ポイントが、冗長コードの組込み無しに提供されることが必要である。

【0012】5. 最小限の影響により新たな装置／データ・タイプへと拡張する機能

新たなマルチメディア・データ標準及び装置タイプは、絶えず作成される。この状況は将来的にも継続するものと思われ、データ伝送機構は新たな装置及びデータ・タイプを支持するように、容易に且つ不断に適応することが必要である。

【0013】6. ストリーミング中における物理メモリに対する要求を最小限に維持する

マルチメディア・アプリケーションにより課せられる重いスループット負荷にも関わらず、データ・ストリーミング中は常にメモリ・リソースの保存が必要である。アプリケーションが物理的なメモリ・マネージャになる必要のないように、バッファ管理サービスが提供されねばならない。しかしながら、いくつかのアプリケーションはマルチメディア・データ・オブジェクトをメモリ内に作成するため、これらのアプリケーションについてはメモリ・バッファの直接的な制御を有する必要はない。この機能はデータ・ストリーミング機構及びその制御サービスにより、不断に支持されねばならない。

【0014】7. ストリーミング中のCPUローディングを最小限に維持する

メモリが慎重に管理されねばならない逼迫したリソースであるように、CPUサイクルは別の重要なリソースであり、マルチメディア・アプリケーションの実行中に浪費されねばならない。データ・ストリーミング機構はハードウェア間のデータ伝送機能（例えばバス・マスター・データ転送）を利用して、CPU負荷を最小に維持する。更に、データ転送機構にデバイス・ポーリング技術

を使用して装置を制御するのではなく、可能な時に割込みにより駆動されるデバイス入出力が利用され、CPUの負荷を軽減する。

【0015】以下の定義は本発明を説明するフレームワークを構成するために提供される。

【0016】データ・ストリーム・・・データ・チャネルを介し、ソース・デバイス（或いはメモリ・アドレス）からターゲット・デバイス（或いはメモリ・アドレス）へ連続的にデータ転送するためのソフトウェア手段。

【0017】ダイナミック・リンク・ライブラリ（DLL）・・・OS/2アーキテクチャでは、16ビット或いは32ビットの実行可能モジュールが特権レベル2または3で実行され、別の実行可能モジュール（例えば、EXE）の外部参照を解決するためにメモリにロードされる。DLLはタスク・コンテキストにおいてのみ実行可能である。

【0018】物理デバイス・ドライバ（PDD）・・・OS/2アーキテクチャでは、16ビットの実行可能モジュールが特権レベル0（リング0）で実行され、典型的には入出力装置のオペレーションを制御する。PDDはタスク・コンテキスト或いはハードウェア割込みコンテキストにおいて実行される。

【0019】ストリーム・・・データをソースから宛先（或いはターゲット）に連続的に転送する。

【0020】

【課題を解決するための手段】本発明は一般化されたデータ・ストリーム転送を改善し、様々な入出力装置間のマルチメディア・データ伝送機構、アプリケーション・メモリ、及び上述の問題を解決し更に上述の要求を満足するように設計されたシステム・メモリを提供する。伝送機構は複数の実行特権レベルを組むマルチタスキング・オペレーティング・システムを拡張することによって実施される。この機構はデータ・ストリーミングと称される連続的な、実時間のデータ転送をサポートする。

【0021】

【実施例】図を参照しながら説明すると、図1は一般的なデータ処理システムを示し、例えばOS/2バージョン2.0などのマルチタスキング・オペレーティング・システム下で動作するパーソナル・コンピュータ10を含み、アプリケーション・プログラムを実行する。コンピュータ10はマイクロプロセッサ12を含み、これはローカル・バス14に接続され、ローカル・バス14はバス・インターフェース制御装置（BIC）16、数理コプロセッサ18、及びSCSI（smallcomputer system interface）アダプタ20に接続される。マイクロプロセッサ12は好適には、80386或いは80486マイクロプロセッサなどの80xxマイクロプロセッサ・ファミリの1つであり、ローカル・バス14はこうしたプロセッサのアーキテクチャに準じる従来のデ

タ、アドレス及び制御ラインを含む。アダプタ20もまたSCSIバス22に接続され、SCSIバス22はCドライブとして指定されるSCSIハード・ドライブ（HD）24に接続される。バス22は他のSCSI装置（図示せず）にも接続可能である。アダプタ20はまたNVRAM30及び読み出し専用メモリ（ROM）32にも接続される。

【0022】BIC16は2つの主な機能を実行する。1つはメモリ制御装置が主メモリ36及びROM38をアクセスする機能である。主メモリ16はダイナミック・ランダム・アクセス・メモリ（DRAM）であり、マイクロプロセッサ12による実行のためのデータ及びプログラムを格納する。ROM38はPOSTプログラム40及びBIOS42を格納する。POSTプログラム40はコンピュータ10がパワーオン或いはキーボード・リセットによりスタートされる時に、標準のパワーオン及びシステムのセルフ・テストを実行する。アドレス及び制御バス37はBIC16をメモリ36及びROM38に接続する。データ・バス39はメモリ36及びROM38をデータ・バッファ41に接続し、データ・バッファ41は更にバス14のデータ・バス14Dに接続される。制御ライン45はBIC16及びデータ・バッファ41を相互接続する。

【0023】BIC16の他の主な機能は、バス14とマイクロ・チャネル（MC）・アーキテクチャに準じて設計される入出力バス44間のインターフェースである。バス44は更に入出力制御装置（IOC）46、ビデオ信号プロセッサ（VSP）48、デジタル信号プロセッサ（DSP）49、及び複数の拡張コネクタ（EC）或いはスロット50に接続される。VSP48は更にビデオRAM（VRAM）60及びマルチブレクサ（MUX）62に接続される。VRAM60はモニタ68の画面上の表示を制御するテキスト及びグラフィック情報を格納する。MUX62は更にデジタル・アナログ変換器（DAC）66、及び、ビデオ機能バス（VFB）に接続可能なコネクタ或いはターミナル70に接続される。DAC66は従来の出力画面或いは表示を提供するモニタ68に接続され、これはユーザーによって観察される。

【0024】IOC46は複数の入出力装置のオペレーションを制御し、これらの装置にはAドライブとして指定されるフロッピ・ディスク・ドライブ72、プリンタ74、及びキーボード76が含まれる。ドライブ72は制御装置（図示せず）及び取り外し可能フロッピ・ディスク或いはディスクケット73を含む。IOC46はまたマウス・コネクタ78、シリアル・ポート・コネクタ80、及びスピーカ・コネクタ82に接続され、これらのコネクタにより種々のオプション装置がシステムに接続される。

【0025】DSP49は更に命令RAM84、データRAM86、アナログ・インターフェース制御装置（AI

C) 88、及びオーディオ制御装置90に接続される。RAM84及び86はそれぞれDSP49が信号処理するために使用する命令及びデータを保持する。オーディオ制御装置90は種々のオーディオ入出力を制御し、また複数のコネクタ92に接続される。これらのコネクタにより様々な装置がシステムに接続可能となる。こうした装置にはヘッドフォン、マイクロフォン、スピーカ、楽器デジタル化インタフェース(MIDI)、及びオーディオ・ライン・イン及びライン・アウト機能を必要とする装置が含まれる。他の様々なマルチメディア装置(MMD)96も、EC50及びアダプタ・カード94を介してシステムに接続可能である。

【0026】メモリ36はシステムで実行される様々なプログラムを格納し、これらのプログラムにはマルチメディア・アプリケーション・プログラム(MMAP)102を含むアプリケーション・プログラム100、及びオペレーティング・システム98が含まれ、後者はsync/stream sub-system(S/SS)104を含む拡張を有する。ここで図1は典型的なマルチメディア・システムを表すが、オペレーティング・システムは汎用目的であり、図1で示される構成とは異なる構成を有するデータ処理システムについても実行或いは制御できるように設計されることを述べておく。本発明は主にS/SS104及びそのオペレーティング・システム100との対話において実施され、これについて次に説明することにする。

【0027】SYNC/STREAM SUB-SYSTEM

【0028】図2を参照すると、マルチメディア・アプリケーション・プログラム102はオペレーティング・システム98の上の層で実行され、マルチメディア制御インタフェース(MCI)を介して、マルチメディア環境における装置を制御するためのMCIコマンドを送信することにより通信する。いくつかの基本コマンドにはpause、play、record、resume、seek、save、set、stopなどがある。こうしたコマンドはオペレーティング・システム98によりメディア・デバイス・マネージャ(MDM)106に転送される。MMAP用のアプリケーションプログラミング・モデルはOS/2 Presentation Manager プログラミング・モデルの論理拡張であり、オブジェクト指向のメッセージング構成体及び手順(call及びreturn)プログラミング・インタフェースの両方を組み込む。MCIはアプリケーション開発者、及び、ビデオ及びオーディオ家庭演奏システムの開発者などのユーザにビューを提供する。オペレーションはメディア装置として知られるメディア情報用のプロセッサを制御することにより実行される。メディア装置は内部或いは外部ハードウェア装置であるか、或いは、低レベルのハードウェア要素及びシステム・ソフトウェア機能を操作することにより、定義された一連のオペレーションを実行するソフトウェア・ライブラリが存在する。複数のメデ

ィア装置をシナリオに含むことが可能であり、これらの装置は再生を同期させるために、グループとして割り付けられて制御される。

【0029】マルチメディア・アプリケーションはシステムの実時間の振舞いにおける2つの面を制御する必要があり、ある装置から別の装置への大量のデータの転送、及び関連する事象の同期がそれに相当する。プログラムにより制御される事象はユーザの直接の制御の下で認識されが必要であり、基本となるシステム機能はこれらの事象が予測可能であり実時間において発生することを容易にし、また保証する。マルチメディア・アプリケーションの開発者は実時間クロックを基本に動作するプログラムを作成すべきである。

【0030】MCIは2つのレベルの動的なリンク・ライブラリ(DLL)を有し、これらはMDM106及びメディア・ドライバを含み、後者はオーディオ・メディア・ドライバ108及びMIDIメディア・ドライバ110を含む。MDM106はメディア装置のリソース管理を提供する。また、MDM106はメディア装置へのアクセスの競合を解決し、アプリケーション開発者にハードウェアに独立なリソースの考え方を提供する。メディア・ドライバは動的なリンク・ライブラリであり、メディア装置の機能を実施する。メディア・ドライバはハードウェア装置のサービス或いはソフトウェアを呼出し、これらの機能を実施する。メディア・ドライバは直接的にはハードウェア装置を制御しない。その代わり、コマンドをストリーム・プログラミング・インタフェース(SPI)112及びsync/streamマネージャ(SSM)114を介して、S/SS104に転送する。SSM114は同期及びストリーミング活動を制御する。SSM114は2つの主な機能を実行し、その第1は連続的な実時間のデータ・ストリーミングを提供するためのデータ・ストリーミングの制御であり、これは本出願の目的である。第2の機能はデータ・ストリームの同期を含み、こうした機能の詳細は上述の関連するアプリケーションにより実施される。

【0031】ストリーム・ハンドラがシステム・カーネル・レベル及びアプリケーション・レベルの両方で必要とされる。いくつかのデータ・ストリームは、ストリーム・ハンドラとリンク0の特権レベルを有する物理デバイス・ドライバとの間の直接接続により最適に制御される。こうしたストリーム・ハンドラはOS/2 Interdevice Driver Communication (IDC)に基づく共通のインターフェースを使用してPDDと通信する。他のデータ・ストリームは特定の物理装置にマップ化可能なデータ・ソース或いはターゲットとは関連せず、DLLによりリンク3の優先レベルで制御される。点線113は一般にどの項目が異なる優先レベルにおいて動作するかを示している。SSM114において、扱っているタスクに応じて、いくつかのルーチンはあるレベルで動作し、他のル

ーチンは別のレベルで動作する。

【0032】各ストリーム・ハンドラはプログラマブルであり、ストリーム・プロトコルに準じてストリーミング可能である。SSM114の観点より、全てのストリーム・ハンドラは同様な責務を負う。各ハンドラは1つ以上のデータ・ストリームに対応するソース或いはターゲットに相当するように設計され、各データ・ストリームはデータを独立に転送する。マネージャ114はストリーム・ソースを適切なストリーム・ターゲットに接続してデータ・フローを維持し、またストリームの終了に際し、種々のリソースをクリーニングする役割を担う。更に、ストリーム・ハンドラはデバイス依存型のモジュールではない。各ストリーム・ハンドラは特定の事前に定義されたタイプのデータのストリーミングを支持するが、データはあるハンドラから次のハンドラに、ハードウェアの振舞いに関する知識無しに転送される。また、オーディオ・ストリーム・ハンドラ134は互換性を有するオーディオ装置のPDDと、完全にハードウェア独立形式で通信する。互換であるためには、PDDは標準オーディオ・デバイス・ドライバ・インターフェースI/O CTLの場合同様、IDCインターフェースに準じることが必要である。図示されるように、ストリーム・ハンドラ114は複数のストリーム・ハンドラ動的リンク・ライブラリ(DLL)116-126と対話する。これらはそれぞれMIDIマッパー、CD/DA、メモリ、分割ストリーム、CD-ROM/XA、及びファイル・システムのストリーム・ハンドラに対応する。ファイル・システム・ストリーム・ハンドラDLL126はファイル・システム130をアクセスするために、マルチメディア入出力(MMIO)マネージャ128を呼出す。

【0033】ストリーム・マネージャ114はまた、ストリーム・マネージャ・ヘルパ133を介して、オーディオ・ストリーム・ハンドラ物理デバイス・ドライバ(PDD)134と対話する。134はストリーム・ハンドラ/PDDインターフェース136及び複数のPDD138-144を介して、選択的に物理装置をアクセスする。ストリーム・マネージャ114はまた、デバイス間通信(IDC)インターフェース146を介してインターフェース136と対話できる。

【0034】図3はデータ・ストリーミング・オペレーションの一般化モデルであり、その詳細は以降の図に示されるフローチャート及びデータ構造に関連して説明される。図3は一般に単一のデータ・ストリーム151を示し、データがストリーム・マネージャ114、及びソース及びターゲット・ストリーム・ハンドラ152及び154の制御により伝送される様子を示す。複数のストリーム・バッファ156がメモリに割当てられ、ストリーミングに使用される。バッファ156はソース・デバイス158からのストリーム・データで充填され、ストリーム・データはターゲット・デバイス160にデータ

を転送することにより空状態となる。データ・ストリーム151はソース・ストリーム・データ153とターゲット・ストリーム・データ155の2つの部分から構成される。ソース・ストリーム・データのデータ経路は、ソース158からソースPDD162及びストリーム・ハンドラ152を介し、バッファ156に至る。ターゲット・ストリーム・データ155のデータ経路はバッファ156からターゲット・ストリーム・ハンドラ154及びターゲットPDD164を介し、ターゲット・デバイス160に至る。ソース・ストリーム・ハンドラ152はソースPDD162をアクチュエートし、PDD162はソース・デバイスのオペレーションを制御する。ターゲット・ストリーム・ハンドラ154はターゲットPDD164をアクチュエートし、PDD164はターゲット・デバイス160を制御する。ソース・ストリーム・ハンドラ152の一般的な目的は、ターゲット・デバイスがスタートする以前にストリーム・バッファ156の少なくとも2つを充填し、ターゲット・デバイスが一度スタートすると、全てのソース・データが転送されるまで、バッファを充填することにより、ターゲット・デバイスよりも先行することである。バッファが充填されると、ターゲット・ストリーム・ハンドラはそこからターゲット・データを獲得し、それをターゲット・デバイスに転送できる。

【0035】メディア・ドライバ108はSPIファンクション或いは呼出を送信し、ストリーム事象通知を受信することにより、SPIインターフェースと対話する。マネージャ114はSPI呼出を解釈し、それに応答して目的の機能を実行する。これはシステム・ヘルパ・コマンド(SHC)をハンドラへ送信し、ストリーム・マネージャ・ヘルパ(SMH)呼出をハンドラから受信することにより、ストリーム・ハンドラと対話して実行される。メディア・ドライバ108はまた、標準I/OCTLコマンドにより定義される制御ファンクションを発行することにより、PDD164を直接制御できる。本発明に関連する主なSPI呼出はSpiCreateStream及びSpiStartStreamであり、それぞれ目的のストリームをセットアップし、続いてデータのストリーミングを開始する。これについては後に図20及び21を参照して更に詳細に説明する。同期されなければならない複数のストリームが存在する場合には、SpiEnableSync呼出が実施され、これについては関連するアプリケーションにおいて詳しく説明されている。

【0036】図4はストリーム・ハンドラ・コマンドSH.C.CREATEが実行される時に、何が発生するかを表す。しかしながら、これを説明する前に、アプリケーション・プログラムに関連して、どのようにそのポイントに達するかを理解することが有益である。第1に、MMA Pはデータ・ストリームを生成するためにMC1呼出を発行する。こうした呼出はOS98により解析され、OSは

メディア・デバイス・マネージャ106に情報を送信する。このマネージャは次に適切なメディア・ドライバを選択する。対称となるドライバはSpiCreateStream呼出をsync/streamマネージャ114に送信するドライバであり、sync/streamマネージャ114は続いてSHC.CREATEコマンドを適切なストリーム・ハンドラに発行する。

【0037】図4はSHC.CREATEコマンドがDLLソース・ストリーム・ハンドラ152により実行されるときの様子を示す。最初にステップ201で、OSの下でスレッドを生成する。スレッドはOSのマルチタスキング機能により制御される。ステップ202ではスレッドに対し、実行されるタスクに適切な優先レベルを割当てる。次にスレッドはステップ203でセマフォー上でブロックされ、“スリープ”状態となる。優先レベルは、OSがスレッドを割当て、実行するレートを制御するために使用される。不連続性が生じる場合には、優先レベルは調整される。

【0038】ここでOS/2の標準オペレーションに準じ、スレッドは個々のタスクとして扱われ、スレッドがブロックされる時、呼出が生成される時、及び復帰が生成される時などにおいて、OSへの復帰を制御する。これによりオペレーティング・システムはマルチタスク方式により、システムにおける他のタスクを実行したり、ストリーミング・スレッドに復帰したりする。また、図には丸で囲まれた数値が矢印で接続されているが、これらはプログラムのフローを示している。例えば、接続印の1が図4と図20で現れる。図20では、接続印は呼出と復帰の両方を含むのに対し、図4の接続印ではルーチンがどこかで呼出されていることを示す呼出の矢印だけを含む。他の接続印についても同様に解釈できる。

【0039】アプリケーション・プログラムにおいて生成されるスタート・コマンドに応答して、マネージャ114はSHC.STARTコマンドを最初にソース・ストリーム・ハンドラに(図4の206)、続いてターゲット・ストリーム・ハンドラに送信する。ソース・ストリーム・ハンドラは、ターゲット・ハンドラがストリーム・バッファに転送されるデータを使用可能になる以前に、当該バッファを充填するために最初に起動される必要がある。こうしたコマンドに応答して、ソース・ストリーム・ハンドラはステップ208でソース・スレッドをブロック解除する。ブロック解除或いは喚起に応答して、ソース・スレッドは次にステップ216で、マネージャ114から空(エンブティ)バッファを要求する。ステップ218でバッファが使用可能でないと判断されると、ステップ220で再びスレッドをブロックする。空バッファが使用可能な場合には、ステップ222でソース・デバイスからデータを読み出し、バッファを充填する。次にステップ224で充填されたバッファをマネージャ114に戻す。ステップ226では更にバッファを充填する必要があるかを判断する。必要がある場合、ステップ

216に分岐し、ステップ216からステップ226に至るループが形成され、このループはステップ226でバッファが充填される必要がないと判断される時にブレークされる。次にスレッドはブロックされる。ストリーミング・オペレーションが一度開始されると、ソース・ファイルの終わりがソース・スレッドが静止するするポイントに達するまで、バッファ充填処理が繰り返される。事象検出は図4のオペレーション・シーケンス内のアスタリスクで示されるポイントであれば挿入可能であり、例えば、キュー(cue)・ポイント検出により、転送されるデータ・レートにより決まる時間を追跡する。事象検出はまた、後に続く数多くのルーチン内でも使用される。

【0040】図5はDLLターゲット・ストリーム・ハンドラをリング3の優先レベルでオペレートする場合を示す。ターゲット・ストリーム・ハンドラはターゲット・データ・ストリームを生成するためにマネージャ114からSHC.CREATEを受信し、最初にステップ230でターゲット・ストリーム・スレッドを生成する。次にステップ232でスレッドに対し優先度を割当て、ステップ234でセマフォー上のターゲット・スレッドをブロックする。ステップ236でマネージャ114がSHC.STARTコマンドを発行することにより、ターゲット・スレッドが喚起されると、ターゲット・スレッドはステップ238でブロック解除され、ステップ240でスレッドは喚起される。ステップ242ではデータ・ストリームからフルのバッファを獲得しようとする。ステップ244で、バッファが獲得されたかを確認し、獲得されていない場合は、ステップ246でターゲット・スレッドをブロックする。フルのバッファが獲得されると、ステップ248でバッファ内のデータをターゲット・デバイスに書き込み、次にステップ250で空バッファをマネージャ114に戻す。ステップ252ではバッファがデータ・ストリームにおける最終バッファであるか、すなわちEOSに達したかどうかが確認される。そうでない場合には、ステップ242に分岐が発生し、フルのバッファを獲得するための処理が繰り返される。ループはストリームの終わりが検出されるまで繰り返され、検出の際にステップ254でスレッドを終了する。

【0041】図6を参照すると、ターゲット・ストリーム・ハンドラが物理デバイス・ドライバにおいてリング0の優先レベルで実施される様子が示される。ステップ256でSHC.CREATEコマンドが受信されると、ステップ258でストリーム・データ構造がセットアップされ、ステップ260ではターゲット・デバイスが初期化される。ステップ261で復帰が行われる。ステップ262でストリームが開始されると、ステップ264でフルのバッファに対する要求がマネージャ114に出される。ステップ266でバッファが提供されたかが判断される。もしそうであれば、ステップ268で入出力書き込み

が開始され、その後ステップ270で復帰する。ステップ272によりエラーが生じていると判断され、その結果、バッファが返却されない場合には、ステップ274でエラーが報告され、次に復帰する。

【0042】バッファへの書き込み及びエンプティ処理により入出力オペレーションが完了すると、ステップ272は割込みを発生し、これはステップ274で割込みハンドラにより処理される。入出力オペレーションがエラー無しに達成されると、ステップ278は空バッファをマネージャ114に戻し、ステップ264に分岐が発生し、処理が繰り返される。入出力オペレーション中にエラーが発生すると、ステップ280でエラーが報告され、ステップ282で復帰する。ソース・ストリーム・ハンドラによるリング0の優先レベル処理は、空バッファを獲得し、フルのバッファを戻す以外は、図6の処理と同様である。

【0043】使用可能なデータ・ファイル及びストリームには2つのタイプがあり、それらは統一或いは非インタリード・ストリーム、及び、分割或いはインタリード・ストリームである。統一データ・ストリーム及びファイルは、例えばオーディオ・データである1つのタイプのデータだけを含む。これに対し分割ストリームは、例えばオーディオ・データとビデオ・データの両方の複数タイプのデータを含む。CD-ROM装置用に一般に生成されるファイル・タイプが例として挙げられる。図7は異なるタイプの例を示す。ファイル286は所与のタイプのオーディオ・データだけを含む統一ファイルを表す。こうしたファイルは複数のバッファB1～B3に読み込まれる。バッファB1にはオーディオ・データの第1のセクションが充填され、バッファB2にはオーディオ・データの次のセクションが充填される。ファイル288は分割或いはインタリード・ファイルを表し、インタリードされたオーディオ及びビデオ・データの複数のセクションが含まれる。セクションは、オーディオが、提示されるビデオに対応するように、対に構成される。こうしたデータはレコードの如く、関連するバッファB4～B6に転送される。すなわち、第1のオーディオ・セクションA1はB4においてオーディオ・レコードAR1に、第1のビデオ・セクションV1はB4のビデオ・レコードVR1に、また第2のオーディオ・セクションA2はB4のAR2となる。次に、セクションV2、A3及びV3がバッファ5にレコードVR2、AR3及びVR3として転送される。バッファB6も同様にAR4及びVR4により充填される。明らかに、この例においては、バッファB4～B6は、各々、3つのレコードを保持するサイズとなる。

【0044】ストリームがマネージャ114により生成されると、複数のバッファは各ストリームに割当てられ、ストリームを管理するために使用されるデータを含むデータ構造が形成される。図8を参照すると、各スト

リームに対し、バッファ・ディレクトリ構造BDSが生成され、これが他のストリームのBDSのリンク・リストに追加される。バッファ制御ブロック(BCB)の連鎖が各ストリームに対して生成され、ストリームには各バッファに対して1つのBCBが割当てられる。図示されるように、BDS1がストリーム#1に対して生成され、3つのBCBすなわちBCB1～BCB3に連鎖される。これらのBCBはそれぞれ関連し、3つのバッファB1～B3に対するポインタがストリーム#1に割当てられる。

【0045】各BDSはストリームに割当てられるバッファの数、ストリームの状態、(分割ストリームにおける)バッファ内のレコードの数に関する情報、及び第1のフルのバッファのBCBに対するポインタ200、空バッファのBCBに対するポインタ202、及び、リスト内の次のBDSに対するポインタ204を記憶するための複数のフィールドを含む。ストリームの異なる状態には、実行中、停止、休止、及びプレロール(prerolled)或いはプライム(primed)がある。後者はオプションであり、ターゲット・デバイスがスタートする以前に、全てのバッファをソースから充填することによりクリック・スタートを提供する。各BCBはバッファ内のバイト数、分割ストリーム・バッファ内のレコードの数、バッファに対するポインタ206、連鎖内の次のBCBに対するポインタ208、及び、BCB連鎖の先頭及び末尾に対するポインタを含む情報を記憶する。

【0046】ストリーム#3及び#4は図7の例を使用して、分割ストリームに対応するデータ構造のバッファリングを表す。BDSが分割ストリーム内の各データ・タイプに対応してセットアップされる。BDS3はオーディオ・データ用に、またBDS4はビデオ・データ用にセットアップされる。バッファB1が3つのレコードにより充填されると、2つのレコード制御ブロックRCB1及びRCB2が生成され、充填されて、オーディオ・レコードAR1及びAR2のB1におけるオフセット及びオフセット長を示す。BCB4はバッファB1に2つのレコードが存在することを示すようにセットされる。並行してRCB3が生成され、ビデオ・レコードVR1のバッファB1におけるオフセット及びオフセット長を示す。BCB7における記録フィールドの数は、そこに記憶される唯一のビデオ・レコードに対応して、1にセットされる。その他のBCB及びRCBについても同様に生成され、分割ストリームに割当てられる残りのバッファに対応して充填される。また分割ストリームに対し、1つのBDSがオーナに指定され、いずれか残りの1つがユーザと見なされる。リンク・リストには各ユーザBDSから関連するオーナBDSを指すポインタ210が含まれる。更に、次に示すポインタが存在する。それらはユーザBCBからオーナBCBを指すポインタ212、及び、BCBからそれに連鎖されるRCBを指

すポインタ214である。

【0047】SSM114はストリーム・マネージャ・ヘルパ(SMH)を使用することにより、ストリーム・ハンドラから呼出される。図9を参照すると、ステップ220で、マネージャ114がバッファ或いはレコードを獲得或いは返却するためにSMH.NOTIFY呼出を受信すると、ステップ222で呼出パラメータが確認され、ステップ224で呼出者がソース・ストリーム・ハンドラか、或いはターゲット・ストリーム・ハンドラであるかを判断する。ソース・ストリーム・ハンドラである場合、ステップ226で、ユーザ或いはアプリケーションがバッファを供給しているかどうかを判断する。そうである場合、ステップ228でGiveBufルーチンを呼出し、このルーチンの復帰に際し、ステップ230は呼出者に復帰する。ステップ226が否定の場合、ステップ232はフルのバッファが返却されているかを判断する。そうである場合、ステップ234でReturnFullルーチンを呼出し、そのルーチンの復帰に際しフルのバッファが返却される場合には、ステップ230を通じて復帰するか、或いは空バッファに対する要求が生じているか判断するためにステップ236が実行される。ステップ236は、ステップ232でフルのバッファに対する要求が出されていないと判断されると、常に実行される。ステップ236が肯定の場合、ステップ238でGetEmptyルーチンが呼出され、ステップ230を介して復帰する。

【0048】ステップ224で呼出者がソース・ストリーム・ハンドラではないと示される場合には、呼出者はターゲットであり、ステップ240で空バッファが返却されているかを判断する。返却されている場合、ステップ242でReturnEmptyルーチンを呼出し、ステップ244で要求がフルのバッファに対するものであるかを判断する。そうでない場合、ステップ230を介して復帰する。そうである場合は、ステップ246でGetFullルーチンを呼出し、その後、ステップ230を介して復帰する。ステップ240が否定の場合、ステップ248でストリームの終わりに達したかがチェックされる。達してない場合、ステップ244に移行し前述のように処理される。一方、達した場合には、ステップ250でバッファをリセットし、ステップ252でアプリケーションにストリームの終わりを報告し、その後、ステップ244に移行する。

【0049】図10はGiveBufルーチン228を示し、ステップ254は提供すべき残りのバッファが存在するかをチェックする。存在しない場合、ステップ256で復帰する。そうしたバッファが存在する場合は、ステップ257で空ポインタがエンティBCBを指示するかをチェックする。指示される場合、ステップ258でMapUserBufferルーチンを呼出す。その後、ステップ260でエラーをチェックする。エラーが発生していない場

合、ステップ262でバッファの属性を保管し、ステップ264でストリームの終わりに達したかが確認される。達していない場合、ステップ266でターゲット待機中フラグがセットされているかがチェックされる。セットされていない場合、ステップ254にループが形成される。ステップ257が否定の場合、ステップ268でソース待機中フラグをセットし、ステップ270でCheckPrerollルーチンを呼出す。ステップ272ではエラーをチェックする。エラーが存在しない場合、制御はステップ254に移行する。ステップ272でエラーが通知される場合、ステップ274でバッファ過剰を示すエラー・フラグをセットし、ステップ256により復帰する。ステップ264が肯定の場合、ステップ276でCheckPrerollルーチンを呼出し、ステップ254に移行する。ステップ266が肯定の場合には、ステップ278で、ターゲット・ストリーム・ハンドラを起動するためにヘルパ・コマンドを送信する。

【0050】図11を参照すると、フルのバッファを獲得するためにReturnFullルーチンが呼出されると、最初にステップ280で、使用可能なフルのバッファが存在するかがチェックされる。存在しない場合、ステップ282で復帰する。フルのバッファが存在する場合、ステップ284でフルのバッファのアドレスに対応するBCBの連鎖を探索し、ステップ286でフルのバッファが見い出されたか、及びソースがそれを所有するかがチェックされる。否定の場合、ステップ288で無効なバッファの返却を示すエラー・フラグをセットし、ステップ282で復帰する。ステップ286が肯定の場合、ステップ290でバッファ或いはレコードが返却されているかが判断される。バッファの場合には、ステップ292でバッファの属性を保管し、ステップ294で保留の放棄ストップが存在するかがチェックされる。存在する場合、ステップ300でCheckStopPendingルーチンを呼出す。ステップ296はステップ300或いはステップ294に続き、ストリームの終わりに達したかを判断する。達した場合、ステップ302でCheckPrerollルーチンを呼び出す。ストリームの終わりに達していない場合は、ステップ298で、ターゲット・ストリーム・ハンドラが起動されるのを待機しているかがチェックされる。そうである場合は、ステップ299でターゲットを起動するためにSHC.STARTルーチンを呼出す。ステップ298が否定の場合、ステップ280ヘループが形成される。

【0051】ステップ290でレコードが返却されると判断されると、ステップ304はレコードがバッファに存在するかを確認する。ステップ306はレコード・ポインタがOKであるかを判断する。そうでない場合、ステップ308で無効なレコードが返却されたことを示すエラー・フラグをセットし、その後、ステップ282で復帰する。ステップ206が肯定の場合、ステップ31

0でレコードの属性を保管し、RCBを割当て、それをRCBリストに追加する。次にステップ312でストリームの終わり及びバッファ状態に対応する最終レコードをチェックし、ステップ294に移行する。

【0052】図12に示されるGetEmptyルーチンの開始に際し、ステップ314は獲得すべきバッファが更に存在するかを判断する。存在しない場合、ステップ316で復帰する。存在する場合は、ステップ318で空ポインタがエンティBCBを指示するかをチェックする。そうである場合、ステップ320でバッファの属性を獲得し、ステップ314にループして戻る。ステップ318が否定の場合、ステップ322で使用可能なバッファが存在しないことを示すソース待機中フラグをターン・オンする。次にステップ324で、CheckPrerollルーチンを呼出す。ステップ326でエラーをチェックする。エラーが存在しない場合、ステップ314へのループ314が形成される。エラーが存在する場合、使用可能なバッファが存在しないことを示すためにエラー・フラグがセットされ、ステップ316で復帰する。

【0053】図13を参照すると、ReturnEmptyルーチンはステップ330で開始され、ここで返却すべきバッファが更に存在するかがチェックされる。存在しない場合、ステップ332で復帰する。バッファが存在する場合、ステップ334でバッファ・アドレスに対応するバッファBCBの連鎖を探索する。ステップ336では、バッファが見い出されて、ターゲット・ストリーム・ハンドラがそれを所有するかが確認される。そうでない場合には、ステップ338で無効なバッファが返却されたことを示すエラー・フラグがセットされる。ステップ336が肯定の場合、ステップ340で、レコード或いはバッファのどちらが返却されるかを判断する。バッファの場合、ステップ342でバッファを空と記し、ステップ346でソース・ストリーム・ハンドラが待機中であるかを確認する。そうでない場合は、ステップ348をスキップする。待機中の場合は、ステップ348でソースに対しSHC.STARTルーチンを呼出す。ステップ344が肯定の場合、次にステップ350はCheckStopPendingルーチンを呼出す。ステップ348或いは350の後にはステップ330が続く。ステップ340でレコードが返却されると判断されると、ステップ352はRCBリストから関連するRCBを除去し、ステップ354で全てのレコードが空状態に戻されたかを判断する。そうである場合、ステップ356でバッファを空と記し、次にステップ344がステップ354或いは356に続く。

【0054】GetFullルーチン(図14)では、ステップ358で更に獲得するバッファが存在するかを判断する。存在しない場合、360で復帰する。存在する場合、ステップ362でフルのポインタがフルのバッファを指示するかを確認する。否定の場合、使用可能なバッファは存在せず、ステップ370でターゲット待機中フ

ラグをセットし、360を介して復帰する。362が肯定の場合、ステップ364でストリームのタイプがノーマル(統一型)か、或いはインターブードかを判断する。統一型の場合、ステップ366でバッファの属性を獲得し、ステップ368で最終バッファの時にストリームの終わり状態(EOS)をセットし、次に360を介して復帰する。インターブード或いは分割ストリームに對しては、ステップ372でバッファ内のフルのレコードを探索する。374のチェックにより発見される場合、ステップ376でレコードの属性を獲得し、ステップ378でバッファ内の最終レコードをチェックし、その後360を介して復帰する。

【0055】図15を参照すると、CheckStopPendingルーチンはステップ380で全てのバッファが返却されたかを確認する。そうでない場合、ステップ384で復帰する。返却された場合は、ステップ386でSMH.ReportEventルーチンを呼出し、“ストリームの停止”を報告し、384で復帰する。

【0056】図16はCHECK_PREROLLルーチンのオペレーションを示し、ステップ388はストリームがプレロール或いはプライムされたか、且つsyncグループに含まれないかを判断する。否定の場合、ステップ390でストリームがマスタであり、全てのスレーブがプレロールされたかをチェックする。否定の場合、392を介して復帰する。ステップ388及び390の結果が肯定の場合、ステップ394でSMH.ReportEventルーチンを呼出し、“ストリーム・プレロール”を報告し、392を介して復帰する。

【0057】図17はリング0及びリング3において使用可能な呼出であるSMH.ReportEventルーチンを示す。ステップ396はイベント・キューがオーバーフローしたかをチェックする。オーバーフローした場合、ステップ397でエラー事象“イベント・キュー・オーバーフロー”を報告し、ステップ398に移行する。また、イベント・キューのオーバーフローが生じていない場合は、ステップ398でその事象を特定の処理に対するイベント・キューに加え、ステップ400でリング3のイベント・スレッドをロック解除し、ステップ402で復帰する。

【0058】図18はMapUserBufferルーチンのオペレーションを示し、ステップ404ではバッファが既にロックされているか、すなわち、バッファが既に自己のリスト内に存在するかを確認するために探索する。バッファがロックされていれば、ステップ406はステップ416に移行し、そのバッファに対応するBCBを新たなBCBリストにコピーし、次に418で復帰する。バッファがロックされていない場合は、ステップ408でバッファをロックしようと試行する。ステップ410では、後者のステップが成功したかを判断する。成功した場合、ステップ420でフィールド情報をそのBCBに

追加し、更にB C Bをバッファ・キューの末尾に追加し、その後4 1 8で復帰する。4 1 0が不成功の場合、ステップ4 1 2で新たにロックされたバッファを割当て、ステップ4 1 4でバッファ・データを新たなバッファにコピーし、その後ステップ4 2 0及び4 1 8を実行する。

【0059】図19では、事象チェックがルーチンの目的のポイントで実施され、ステップ4 2 2ではスレッドをブロックする。スレッドがブロック解除されると、EventRouterルーチンがステップ4 2 4で呼出され、このルーチンは最初にステップ4 2 6で事象に対応するイベント・キューをチェックし、次にステップ4 2 8で事象が取り出される(dequeue)。ステップ4 3 0では事象を取り扱うアプリケーション・イベント・ルーチンが呼出される。次にステップ4 3 2で、更に処理する事象が存在するかがチェックされ、存在する場合は、ステップ4 2 8に戻る。全ての事象が処理されると、ステップ4 3 4でスレッドがブロックされる。

【0060】図20を参照すると、アプリケーション・プログラムがストリームを生成すると、SpiCreateStream API呼出が実施され、制御はsync/streamマネージャ1 1 4に移り、マネージャ1 1 4はステップ4 3 6でSHC.CREATEコマンドを使用して、ソース・ストリーム・ハンドラを呼出す。次にステップ4 3 8で、SHC.CREATEを使用して、ターゲット・ストリーム・ハンドラが生成される。ステップ4 4 0ではストリーミング・プロトコル制御ブロック(図示せず)にリストされるストリーミング・パラメータが獲得され、パラメータにはバッファ数及びサイズ、データ・ストリーム・タイプ、バッファ当たりの最大レコード数、一定のデータ・ストリームを維持するために必要な最小バッファ数、ソース・ストリーム・ハンドラを起動するために必要な空バッファ数、ターゲット・ストリーム・ハンドラを起動するために必要なフル・バッファ数などが含まれる。次にステップ4 4 2で、ソース・ストリーム・ハンドラにストリーム・パラメータを通知し、ステップ4 4 4で、ターゲット・ストリーム・ハンドラにストリーム・パラメータを通知する。ステップ4 4 6ではストリームをセットアップし、ストリーム・バッファを割当てる。その後、ステップ4 4 8でアプリケーションに復帰する。

【0061】図21を参照すると、アプリケーション・プログラムがストリーミング開始呼出を発行すると、マネージャ1 1 4はSpiStartStream API呼出を受信し、ステップ4 5 0がSpiStartStreamルーチンを実行し、これは最初にステップ4 5 2でストリームが休止状態かをチェックする。休止状態でない場合、ステップ4 5 4でストリームがブレロールされているかを判断する。されていない場合、ステップ4 5 6で、SHC.STARTコマンドによりソース・ストリーム・ハンドラを呼出し、ソース・スレッドをブロック解除する。次にステップ4 5 8で、

ストリームがsyncグループにおけるスレープ・ストリームかをチェックする。そうでない場合、ステップ4 6 0でアプリケーション・プログラムに復帰する。そうである場合は、ステップ4 6 2でステップ4 5 2に分岐し、各スレープ・ストリームに対応する処理を繰り返す。ステップ4 5 2が肯定の場合、ステップ4 6 4でSHC.STARTコマンドにより、ソース・ストリーム・ハンドラを呼出してスレッドを解除し、次にステップ4 6 6で、ターゲット・ストリーム・ハンドラを呼出し、ブロック解除する。ステップ4 6 6はまた、ステップ4 5 4の肯定結果にも続く。次に、ステップ4 5 8がステップ4 6 6に続いて実施される。

【0062】図22はイベント・データ構造を表す。各処理に対し、イベント・キューを指示する処理制御ブロックPCBが生成される。図示のように2つのPCBすなわちPCB1及びPCB2が存在し、それぞれ処理#1及び#2に対応してセットアップされる。各PCBはプロセスID、イベント・スレッドID、関連するイベント・キューの先頭を指示するポインタ4 7 2を含む。イベント・キューはヘッダ4 7 4を含み、各ヘッダはフラグ及びユーザ・イベント制御ブロックEVCB4 7 8を指示するポインタを含む。こうしたブロックは拡大して示されるように、イベント・タイプ、イベント・フラグ、ストリーム・ハンドラID、そのステータス、及びユーザ・パラメータを含む。イベント・データ構造内の情報はイベント・ルータ・ルーチン(図19)により使用される。

【0063】図23はSpiEnableルーチン4 8 0のオペレーションを表し、ステップ4 8 2では呼出すストリーム・ハンドラを決定し、ステップ4 8 4でステップ4 8 2で識別されたソース或いはターゲット・ハンドラを呼出す。ステップ4 8 4の後にはスキップ4 8 6が続く。

【0064】図24はDLLの初期化時に生成されるアプリケーションからの最初のSpi API呼出により実行される内容を表す。ステップ4 8 8でEventRouterルーチンを呼出すことによりイベント・スレッドを生成する。ステップ4 9 0でイベント・スレッドの優先度をセットし(リング3または0)、ステップ4 9 2でsync/streamマネージャを呼出し、ステップ4 9 4で復帰する。

【0065】図25は時間キュー(cue)・ポイント事象と称される事象の例を示し、ストリーム・ハンドラ特にターゲット・ストリーム・ハンドラ内に配置される。ステップ4 9 6は現行の時間を決定する。次にステップ4 9 8は、こうした時間において発生する事象に対応するイベント・キュー或いはリストを探索する。ステップ5 0 0は時間の一一致をチェックする。一致が存在しない場合、ステップ5 0 4で復帰する。各一致に対し、ステップ5 0 2でSMH.REPORTEVENTの呼出によりその事象を報告する。

【0066】以下本願発明の内容を要約して述べる。

【0067】本発明はいくつかの独自の特徴を有し、これらは新規なものであり、従来、アプリケーション或いはオペレーティング・システム・ソフトウェアにおけるマルチメディア・データ伝送には組み込まれていなかつた。本発明の独自の特徴としては次の点が含まれる。すなわち、アプリケーション・バッファリング・オプションによる中央バッファ管理；2レベルのデータ・ストリーム・ハンドラのサポート；インタリーブド・データ・ストリームのサポート；及び、データ・ストリーム事象の検出及び通知である。これらの独自の特徴は、存在する物理デバイス・ドライバを超越して完全にデバイスに独立な形式により実施される。この解決においては、更に、データ・タイプはマルチメディア・システム拡張におけるモジュールに依存しない。

【0068】本発明の一般化したデータ・ストリーミング機構はデバイス独立型のマルチメディア・データ伝送を提供し、IBMのバージョン2.0のオペレーティング・システム／2(OS/2)に対するマルチメディア拡張として設計される。この機構は実時間という条件を具備しつつ、記憶及び／或いは検索のためのデータの連続的な転送を支援する。例えば、大量のデジタル化オーディオ・データ・ファイルはハード・ファイルから読み出され、データを連続的にアナログ増幅回路を有するデジタルアナログ変換器(DAC)装置に転送することにより演奏されねばならない。

【0069】データ・ストリーミングの実施例の次の4つの主な特徴は、本発明の中心を成すものである。また、これらはマルチメディア・コンピュータ・システムにおける重要な貢献を代表するものであり、前述のセクションで述べられた全ての問題を解決するものである。

【0070】アプリケーション・バッファリング・オプションによる中央バッファ管理

【0071】データ・ストリーム・リソース管理は中央バッファ・プールに対する／からのバッファ割当を制御し、複数のストリーム・ハンドラの間でバッファを共用し、また動的なバッファのロック／ロック解除を制御する。更にデータ・ストリームは、アプリケーションにより割当てられ管理されるバッファを使用し、特にこれらはアプリケーションが、ストリームを介して転送されるデータの実時間生成を制御している場合に使用される。

【0072】一連のバッファ管理ルーチンは、データ・ストリームが生成される時に、割当てられたバッファ・プールからバッファを獲得しまた返却する機能を提供する。これらのバッファ用のメモリは割当てられ、この時にロック・ダウンされる。バッファのロック・ダウンは、データのストリーミング中に、システムがバッファを含むメモリをスワップ・アウトしないことを保証するために必要である。

【0073】処理されるデータが連続的なストリームと

して処理されることを保証するために、ストリームに関連するストリーム・プロトコル制御ブロックがセットアップされ、これはバッファを割当てるために使用される情報(すなわちバッファ数、バッファ・サイズ、及びストリーム・ハンドラを起動する時を示す値)を含む。ストリームされるデータのタイプによって、バッファの量及びサイズを最適に設定しなければならないためこの情報は重要である。なぜなら、データの連続的なストリーミングの継続、及びシステム・リソースが過度に(すなわち実際に必要なメモリ量以上にメモリを割当てる)使用されないことを保証する必要があるためである。メモリの割当と共に、ストリーム・ハンドラを起動するために使用される値は、ソース・ストリーム・ハンドラにより十分なバッファが充填され、その結果ターゲット・ストリーム・ハンドラが起動可能であり、かつ、実時間データ・ストリーミングを発生するために十分なデータがバッファされていることを保証する。

【0074】中央バッファ管理ルーチンは、ソース・ストリーム・ハンドラがバッファ・プールから空バッファを獲得し、入力装置からのデータ(例えばハード・ディスク上のファイルからのデータ)を充填し、それによってフルになったバッファをバッファ・プールに返却することを許可する。ターゲット・ストリーム・ハンドラはバッファ・プールからフルになったバッファを獲得し、このバッファを出力装置(例えばスピーカにより演奏されるオーディオ装置へデータを転送する)に送信し、その結果空になったバッファをバッファ・プールに返却する。

【0075】バッファ管理ルーチンの独自の特徴は、バッファ内でレコードを扱う機能である。この機能によりインタリーブド・データは最小のデータ転送により処理される。データは一度だけバッファに転送され、続くアクセスは個々のデータ(すなわちオーディオ及びビデオ・データなど)を含むバッファ内のレコードを介して行われる。

【0076】バッファ管理ルーチンの他の独自の特徴は、ソース・ストリーム・ハンドラがその固有のバッファをバッファ・プールに提供することを可能とし、その後、このバッファは要求により、ターゲット・ストリーム・ハンドラに提供される。バッファが提供されると、これらはその時点でメモリにロック・ダウンされ、これによりスワップ・アウトされることはない、また、データ・ストリームの期間中に、全システム・メモリ・リソースを消費することはない。

【0077】次に典型的なストリーム・ハンドラがファイルからデータを読み出し、ストリームをオーディオ出力装置に転送するためのシナリオを示す。

【0078】1. アプリケーションがデータ・ストリームを生成する。バッファが割当てられ、この時ロック・ダウンされる。

2. アプリケーションがデータ・ストリーミングを開始する。

3. ソース・ストリーム・ハンドラが
-空バッファを獲得する。

-入力装置からバッファを充填する。
-フルのバッファを返却する。

-データの終わりに遭遇するまで、上記3ステップをループする。

4. ターゲット・ストリーム・ハンドラが
-フルのバッファを獲得する。

-バッファを出力装置に排出する。
-空バッファを返却する。

-データの終わりに遭遇するまで、上記3ステップをループする。

【0079】こうした一連のバッファ管理ルーチンは、ソース・ストリーム・ハンドラからターゲット・ストリーム・ハンドラへの連続的なデータ・フローの発生を保証する機能を提供する。

【0080】2レベルのデータ・ストリーム・ハンドラの支持

【0081】異なる保護レベルのデータ・ストリーム・ハンドラに対し、同一のシステム・サービスを提供し、システム・ハンドラがOS/2物理デバイス・ドライバ（保護リング0）或いはOS/2動的リンク・ライブライ（保護リング3）として開発されることを可能とする。

【0082】OS/2アーキテクチャは80286を基本とするインテル・ファミリのプロセッサを支持し、これには80386DX、80386SX、及び80486などが含まれる。これらのプロセッサはリング0から3までの一連の特権レベル或いはリングの提供を基本とする保護機構を共有する。各レベルにおいて、異なるオペレーションが、システムの保全性に影響を与える相対的リスクに準じて許可される。特権レベル0或いは“リング0”では、全てのオペレーションは許可される。このレベルは典型的にはシステム・コードの実行用に確保される。アプリケーションはリング3で実行され、特権命令の実行の試行はトランプを生成し、これはリング0のオペレーション・システムにより扱われ、典型的にはアプリケーションの終了に帰結する。

【0083】リング0のソフトウェアの開発は、リング3のソフトウェアの開発に比較して困難であり、またリスクを伴う。これはリング0で活動状態のシステム保護機構の相対的な欠如による。リング0のコードのモジュール開発、テスト、及びそこからエラーを除去するためには、カーネル・デバッガを含む特殊なツールが必要となる。その結果、リング0のコードの特殊な性質を必要としない機能、例えばハードウェア割込みのコンテキストにおいて実行される機能などは、典型的にはリング3で実行されるように作成される。リング3におけるデバ

ッグはより簡単であり、リング3のコードにより生成されるルーチン・エラーは、リング3で実行可能な全てのコードに対して提供される保護機能により、一般には他の処理に影響を及ぼすことはない。

【0084】通常、開発者は、リング0の特権の持つ特殊な性質により目的的機能が提供される時以外は、全てのコードをリング3で実行可能なように開発することを好む。しかしながら、OS/2のリング0のモジュールの開発においては、使用可能なシステム・サービスの数及び柔軟性は非常に制限されている。リング0の実行に応じてロード可能なモジュールのタイプは、唯一、物理デバイス・ドライバ（PDD）である。PDDにとつて唯一使用可能なシステム・サービスは、OS/2カーネルにより提供されるデバイス・ヘルプ（DevH1p）機能である。これらの同一のDevH1pインターフェースはリング3のモジュールでは使用できず、全体的に異なるセットのシステム・サービス・インターフェースを使用してプログラムされねばならない。

【0085】ここでオペレーティング・システムの拡張が導入される時に、常に問題が生ずる。新たなサービスがリング3に限り提供される場合、広範囲に及ぶデバイス制御機能が妨げられる。なぜなら新たなサービスを呼び出すモジュールについても、特権レベル0命令を実行することはできなかったり、或いはハードウェア割込みコンテキストにおいて実行できないからである。新たなサービスがリング0に限り提供される場合、全ての開発者は、彼らが特権レベル0の命令をアクセスする必要がある無しに問わらず、リング0のモジュールを作成する困難を被ることになる。

【0086】好適な実施例としては、データ・ストリーム制御サービスを提供するOS/2の拡張である。これらのサービスはOS/2のDLL及びPDDの対により構成されるSync/Streamマネージャにより提供される。データ・ストリームはSync/Streamマネージャにより提供されるサービスを呼び出す1つ以上のストリーム・ハンドラにより制御される。

【0087】ストリーム・ハンドラの開発者に対して最大の柔軟性を提供し、リング0のコードの急速な拡張を阻止するために、Sync/Streamマネージャにより同一のサービスがリング0及びリング3の両方において提供される。これはすなわち、リング0の特権を厳格に要求するこれらのストリーム・ハンドラだけが、OS/2のPDDとして作成され、他の全てはリング3で実行されるOS/2のDLLとして作成されることを意味する。両方の特権レベルにおいて同一のインターフェースが使用可能ため、ストリーム・ハンドラの開発者はデータ・ストリームを制御するために、1つのセットのインターフェースだけを使用して、コードを作成できる。

【0088】Sync/Streamマネージャは重複するサービス・ルーチンを生成することなしに、リング0及びリン

グ3の両方のモジュールに対し、同一のサービスを提供する。各機能に対して2つの入力ポイントが提供され、一方はリング3 (Sync/Stream マネージャDL対応) であり、他方はリング0 (Sync/StreamマネージャPD D対応) である。これらの入力ポイントの各々は共通のサブルーチンにベクトル指向される。共通のコードが、呼出される特定の機能に最適となるように、リング0或いはリング3で実行される。

【0089】インタリーブド・データ・ストリームの支持

【0090】この機構により、単一のデータ・ストリーム・ソースは、データを複数のターゲットに転送することが可能となる。これは分割ストリームと称される。例えば、ある装置はオーディオ、イメージ、及びテキストのデータ・タイプを、単一のフォーマットされたデータ・トラックにインタリーブする。分割ストリーム・ハンドラは、単一のインタリーブド・データ・バッファから各データ・タイプを抽出し、各データ・タイプを独立に転送し、各ストリームを適切なターゲット・デバイスに引き渡すために使用される。例えば、CD-ROM XA はウェーブオーディオ、ビデオ・データ及びテキスト・データを同一のデータ・バッファ内にインタリーブできる。分割ストリーム・ハンドラは、データをCD-ROM XA 装置用の複数のストリームに分割するために使用される。分割ストリーム・ハンドラにとって、この結合されたデータを1つのバッファに読み出し、次にデータをコピーすることなく、ビデオ・データをビデオ・ストリームに挿入し、またウェーブオーディオ・データをウェーブオーディオ・ストリームに挿入することが、より効率的である。ウェーブオーディオ・データ・ストリームは次にウェーブオーディオ・ストリーム・ハンドラにより、またビデオ・ストリームはビデオ・ストリーム・ハンドラにより消費される。ウェーブオーディオ・ストリーム・ハンドラはデータ・ストリームのオーディオ部分だけを認識し、またビデオ・ストリーム・ハンドラはストリームのビデオ部分だけを認識する。

【0091】データ・ストリームは複数の分割ストリームに分割される。分割の数はシステムのリソースによってのみ制限される。別々に分割されたストリームの各々は分割ストリームである。異なるデータを分析し決定する作業は、ソース・ストリーム・ハンドラの役割であり、実際には空バッファをソース・デバイスからのデータにより充填することを行う。ソース・ストリーム・ハンドラは、どのデータがどのストリームに所属するかを決定し、次に、ポインタをバッファの個々の"レコード"に返却しなければならない。各レコードは、ある"分割"ストリームに特定なデータにより充填されるバッファの一部に相当する。これらのレコードはSync/Stream マネージャに返却され、どのストリームに所属すべきかを示す。Sync/Stream マネージャは、次に、バッファを共用

する分割ストリームの1つに対応して設けられたそれぞれのバッファ/レコード・キューに、これらのレコードをキュー待機させる。

【0092】このタイプの分割ストリームは以下に示すようにセットアップされる。複数のストリームが生成されなければならず、各々はSync/Stream マネージャに対するSpiCreateStream API 呼出により実行される。これらのAPI呼出の第1は、通常のストリームの生成呼出であり、hstreamBufパラメータはNULLにセットされる。第1のストリームのバッファを共用する必要のある他の続くSpiCreateStreamAPI呼出のそれぞれは、hstreamBuf パラメータにおけるそのストリームの制御をパスしなければならない。これはSync/Stream マネージャにより使用される機構であり、これらの分割ストリームによって第1のストリームのバッファを共用するためのものである。ソース・ストリーム・ハンドラはまた、SHC_CREATE呼出によりこれらのパラメータを受信し、この種のタイプのストリーミングを支持することが必要である。SPCBにおけるSPCBBUF_INTERLEAVEDフラグはソース・ストリーム・ハンドラによりセットされ、分割ストリーミングを実行可能かどうかを示す。実行できない場合、SpiCreateStream がそのバッファを使用する試みは拒否される。分割ストリーム機構は1つのソース及び複数のターゲットに対応して機能する。

【0093】データ・ストリーム事象の検出及び通知

【0094】ストリーム事象はデータ・ストリームにおける状態の特定の変更を識別する。データ・ストリーム・ハンドラは特定の事象を検出し、その事象を無視するか、或いはその事象が発生したことを通知する。この通知は別のストリーム・ハンドラ或いはアプリケーション・プログラムにより受信される。

【0095】アプリケーションはデータ・ストリーミング・ルーチンに対し、(イネーブル/ディスイネーブル事象を介し) どの事象が検出されるべきかを知らせることができる。これらの事象は典型的にはエラー検出の通知、ストリーム・キュー時間事象、及びデータ・キュー事象などである。また、ストリーム・ハンドラはデータ・ストリームの実時間性を維持するために実行される特定の活動を保証するデータ・ストリームの状態を通知される必要がある。このために、これらの事象はアプリケーション或いはストリーム・ハンドラ・イベント・ルーチンに適時(実時間で) 送信されて、最小時間が経過する以前に適切な活動が実行される必要がある。この事象の報告は、ストリーム・ハンドラがリング0或いはリング3で実行されているという事実によって若干複雑になる。

【0096】事象に関するリング3のストリーム・ハンドラからリング3のストリーム・ハンドラ或いはアプリケーションへの報告は、登録されたイベント・ルーチンがスレッドによる事象報告のコンテキストにおいて呼出

されるという点で、直接的である。しかしながら、リング0のストリーム・ハンドラがリング3のストリーム・ハンドラ或いはアプリケーションに対して事象を報告する必要が生じた場合には、リング0の割込み時間中にストリーム・ハンドラが事象を検出した場合、この時間内に直接リング3のイベント・ルーチンを呼出す方法が存在しないという点で問題が生ずる。この問題を解決するために、最小のタスク切替えオーバーヘッドを有し、実時間環境において事象を十分な速度で報告できるようにディスパッチされる新たなタイム・クリティカル・スレッドが生成された。この新たなスレッドは実行中は割込み不可である。各アプリケーション処理に対し、この新たなスレッドが生成され、報告される事象をモニタする。事象報告するための事象シーケンスは以下のようにある。

【0097】1. 事象が割込み時間において、リング0のストリーム・ハンドラにより検出される。
2. 事象はイベント・キューに配置され、新たなタイム・クリティカル・スレッドがブロック解除されて実行される。これはリング3のアプリケーションのコンテキストにおいて実行される。
3. タイム・クリティカル・スレッドはアプリケーションのコンテキストにおいて制御を獲得し、イベント・キューから事象を取り出し、アプリケーション・イベント・ルーチンを呼出す。

【0098】この事象シーケンスは最小時間により実行され、それによりアプリケーションは活動を実時間で実行でき、また見た目にもより優れたアプリケーションが指向される。

【0099】当業者においては、本発明の精神及び範囲を逸脱することなく、構成及び詳細における多くの変更が可能であることが理解されよう。

【0100】

【発明の効果】以上説明したように、本発明によれば、マルチタスキング・オペレーティング・システムの制御の下で、実時間において連続的に大量のデータを伝送することを可能とするマルチメディア・システムが提供できる。

【図面の簡単な説明】

【図1】本発明を実施するデータ処理システムのブロック図である。

【図2】図1のシステムにおいて実施されるsync/streamサブシステム・アーキテクチャのブロック図である。

【図3】データ・ストリーミングの一般化モデルを表すブロック図である。

【図4】ソース・ストリーム・ハンドラ・オペレーションの流れ図である。

【図5】ターゲット・ストリーム・ハンドラ・オペレーションの流れ図である。

【図6】ターゲット・ストリーム・ハンドリング・オペ

レーションの流れ図である。

【図7】統一及び分割データ・バッファリングを表す図である。

【図8】バッファリング・データ構造を示す図である。

【図9】sync/stream バッファ管理オペレーションをインターロックする流れ図である。

【図10】sync/stream バッファ管理オペレーションをインターロックする流れ図である。

【図11】sync/stream バッファ管理オペレーションをインターロックする流れ図である。

【図12】sync/stream バッファ管理オペレーションをインターロックする流れ図である。

【図13】sync/stream バッファ管理オペレーションをインターロックする流れ図である。

【図14】sync/stream バッファ管理オペレーションをインターロックする流れ図である。

【図15】sync/stream バッファ管理オペレーションをインターロックする流れ図である。

【図16】sync/stream バッファ管理オペレーションをインターロックする流れ図である。

【図17】sync/stream バッファ管理オペレーションをインターロックする流れ図である。

【図18】sync/stream バッファ管理オペレーションをインターロックする流れ図である。

【図19】sync/stream バッファ管理オペレーションをインターロックする流れ図である。

【図20】sync/stream マネージャ・ストリーム生成オペレーションの流れ図である。

【図21】sync/stream マネージャ・ストリーム開始オペレーションの流れ図である。

【図22】事象データ構造を示す図である。

【図23】事象許可オペレーションの流れ図である。

【図24】オペレーションがアプリケーション・プログラムからsync/streamマネージャに移行する様子を示す流れ図である。

【図25】事象検出オペレーションの例を示す流れ図である。

【符号の説明】

22... SCSIバス

24... SCSIハード・ドライブ (HD)

37... 制御バス

39... データ・バス

41... データ・バッファ

48... ビデオ信号プロセッサ (VSP)

49... デジタル信号プロセッサ (DSP)

60... ビデオRAM (VRAM)

62... マルチプレクサ (MUX)

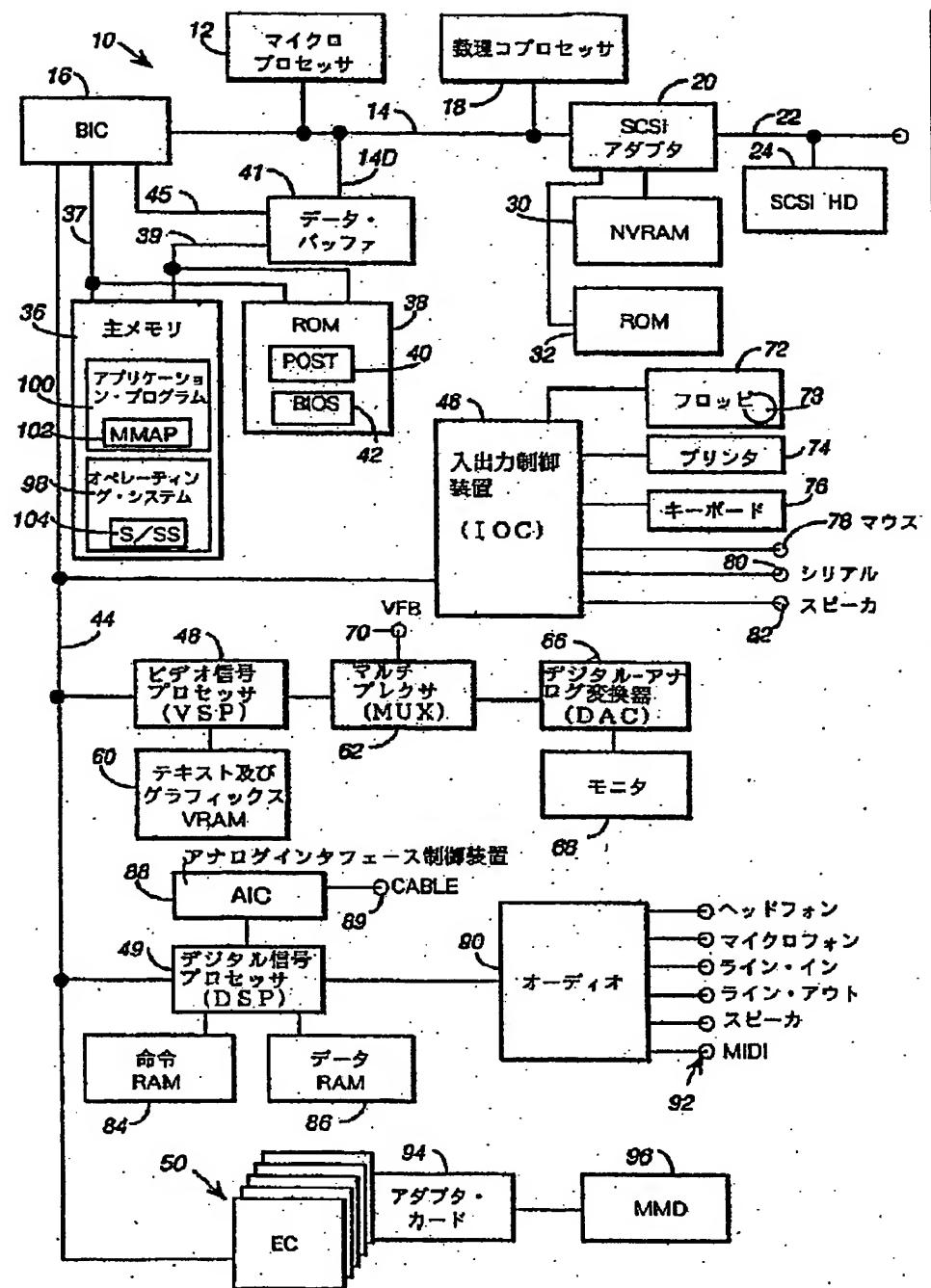
66... デジタル-アナログ変換器 (DAC)

88... アナログ・インターフェース制御装置 (AI)

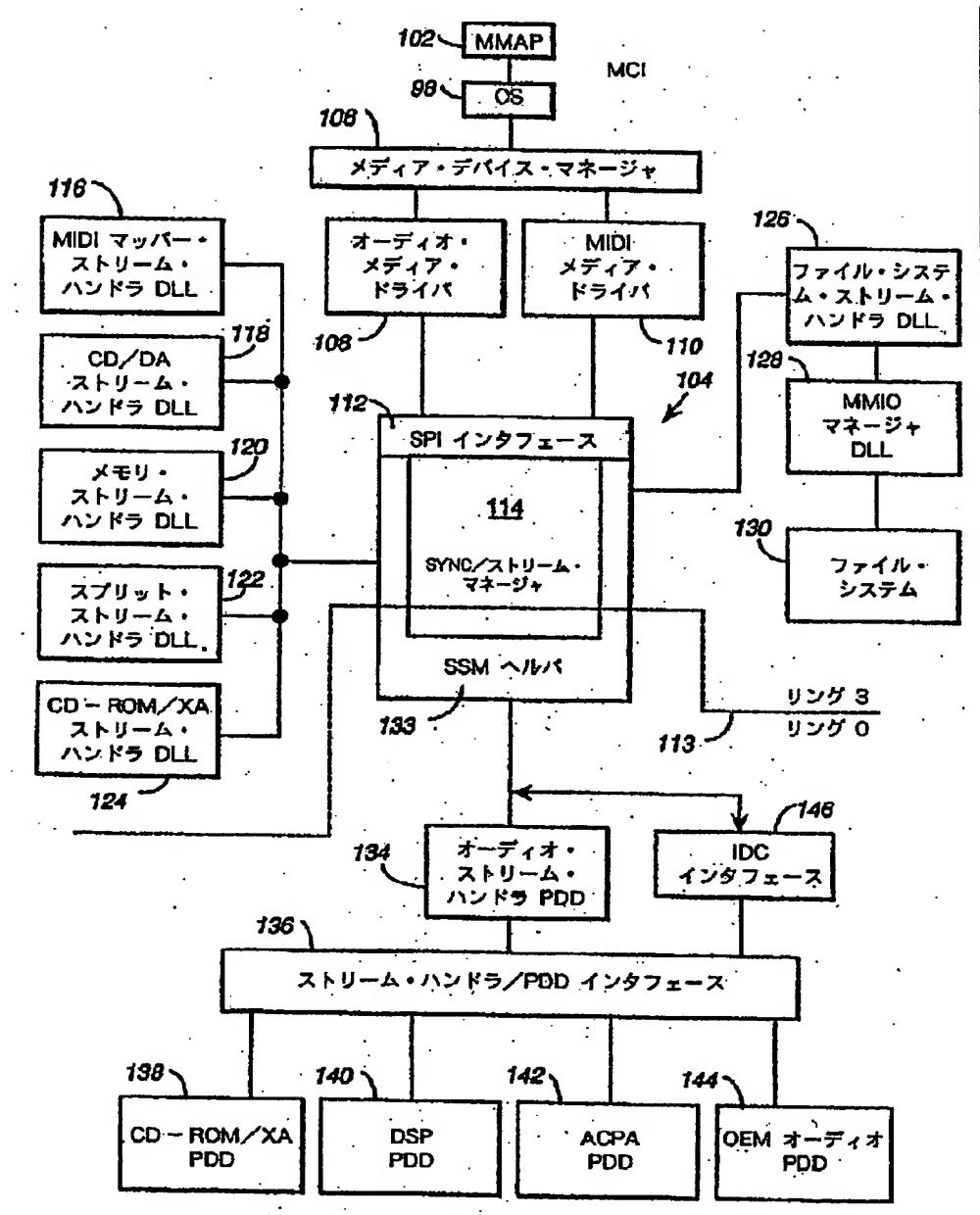
C)

90 . . . オーディオ制御装置	グラム (MMA P)
94 . . . アダプタ・カード	106 . . . メディア・デバイス・マネージャ (MD M)
96 . . . マルチメディア装置 (MMD)	110 . . . M I D I メディア・ドライバ
98 . . . オペレーティング・システム	116 . . . 動的リンク・ライブラリ (D L L)
100 . . . アプリケーション・プログラム	133 . . . ストリーム・マネージャ・ヘルパ
102 . . . マルチメディア・アプリケーション・プロ	

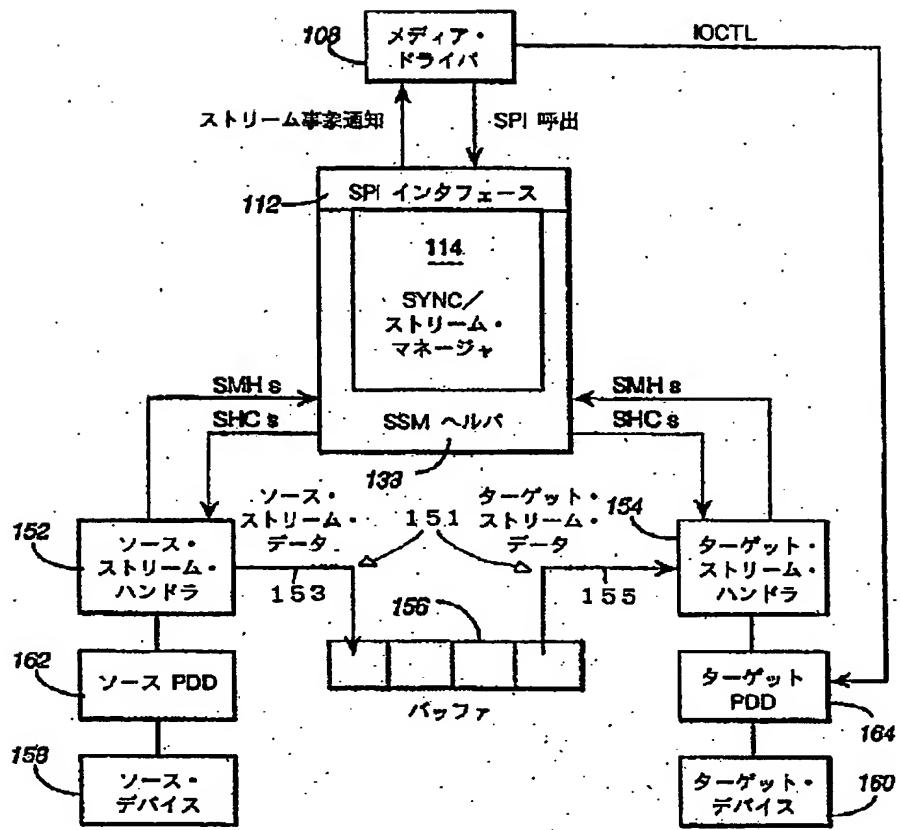
【図1】



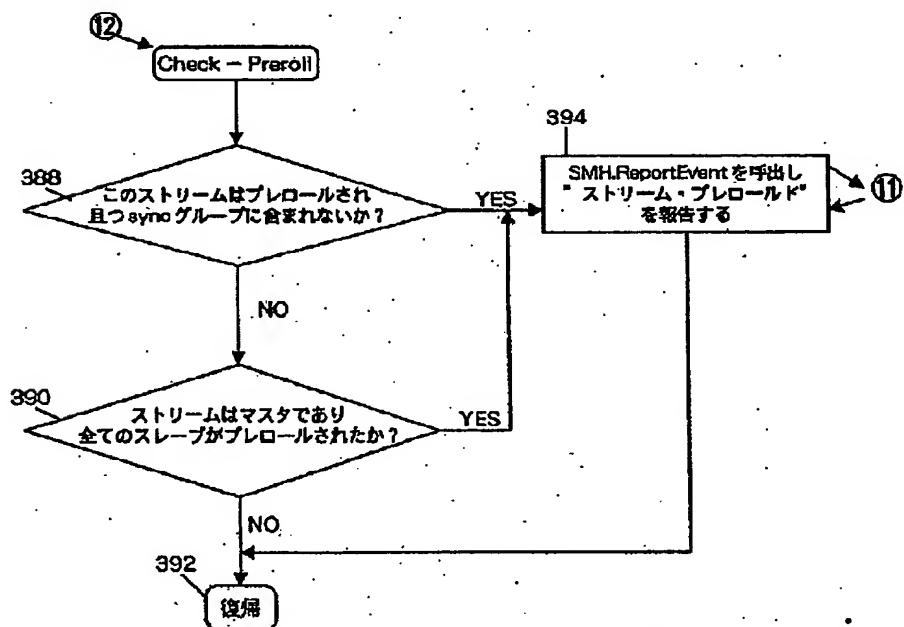
【図2】



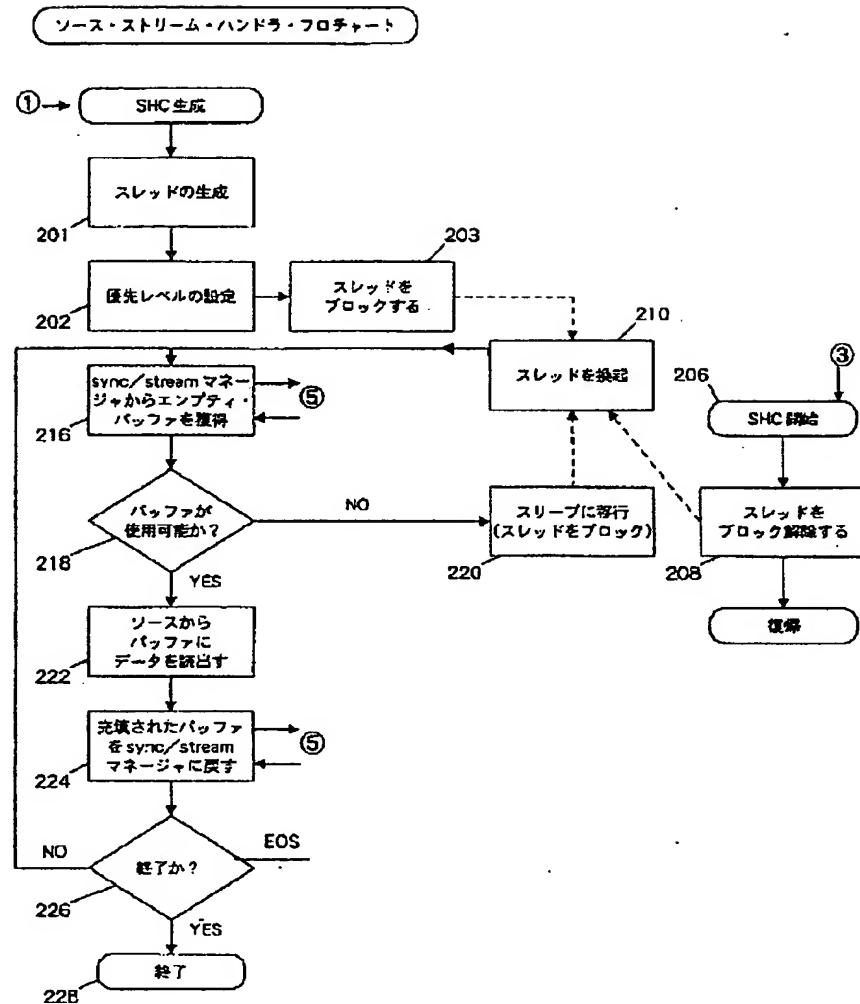
【図3】



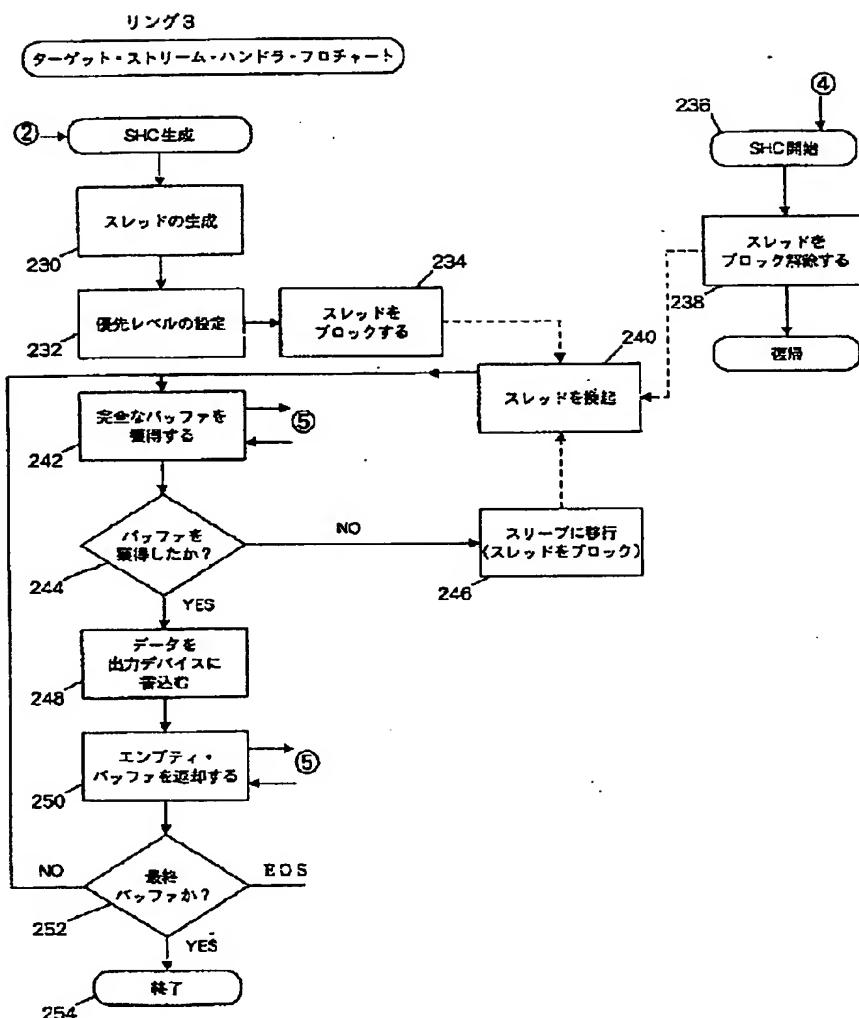
【図16】



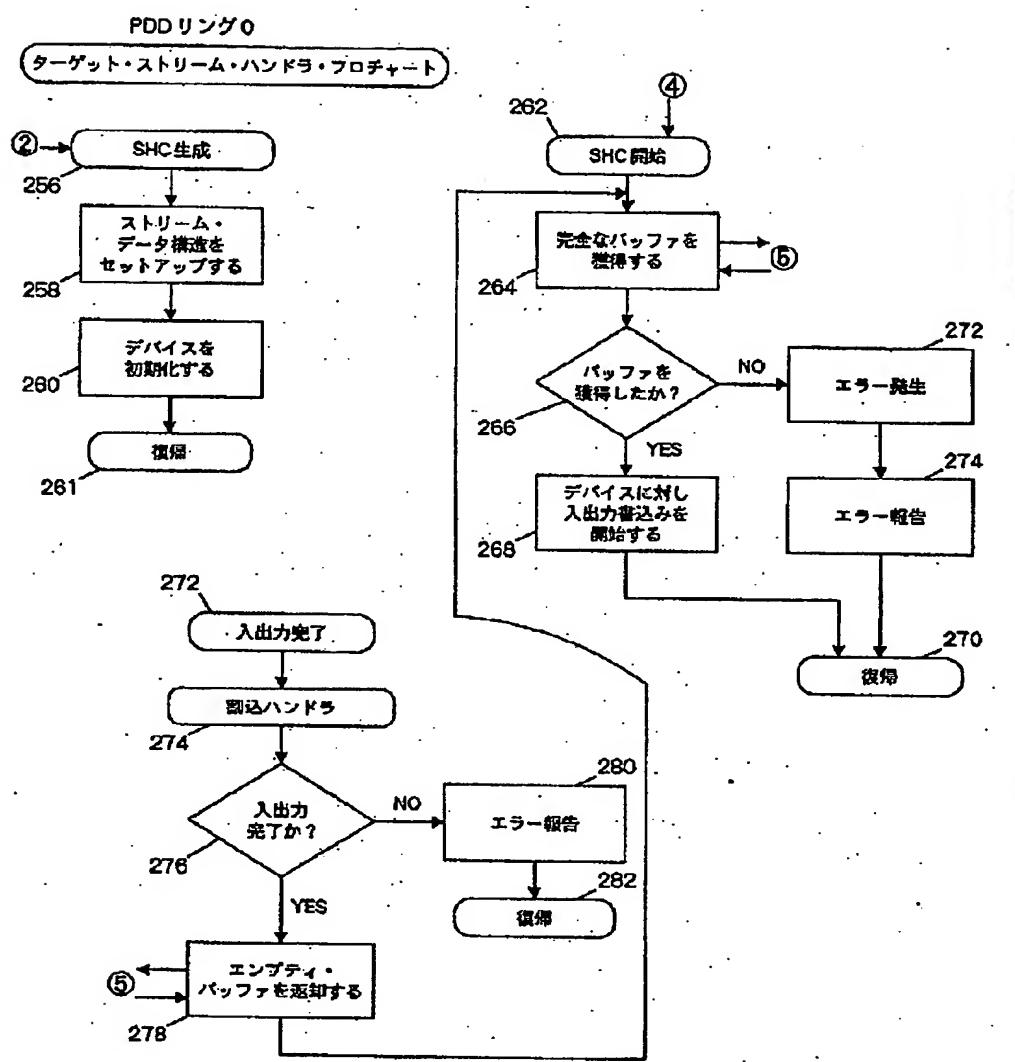
【図4】



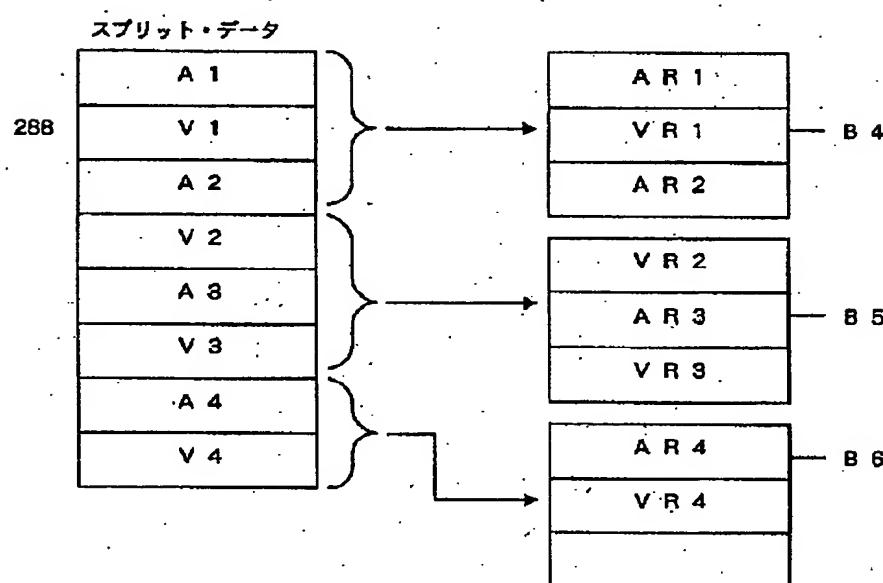
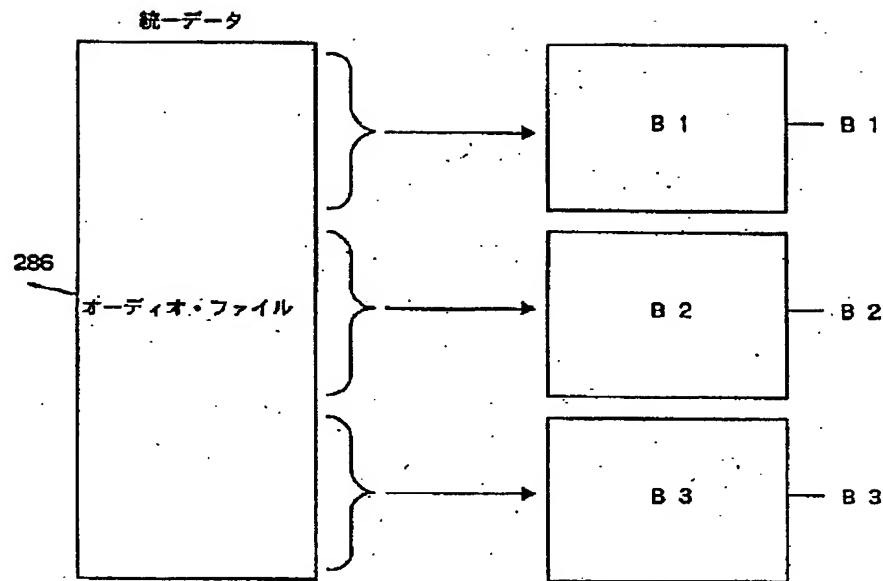
【図5】



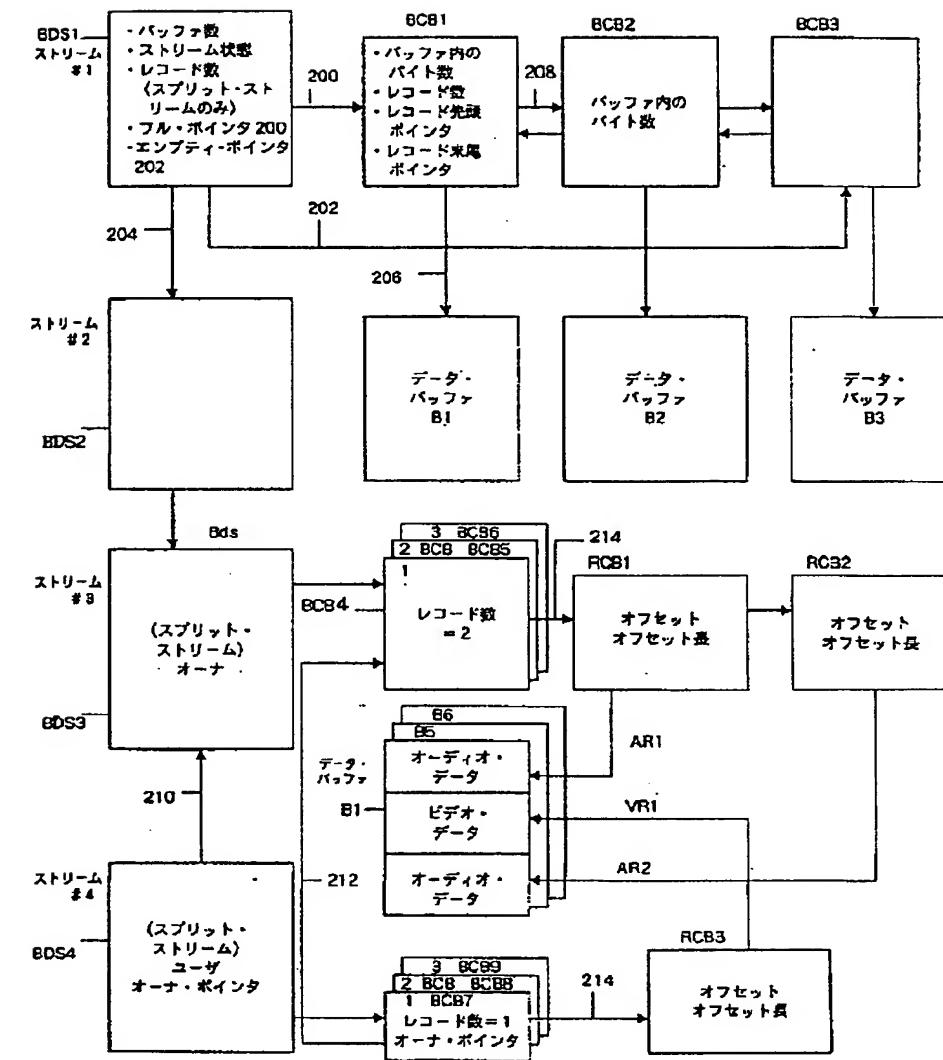
【図6】



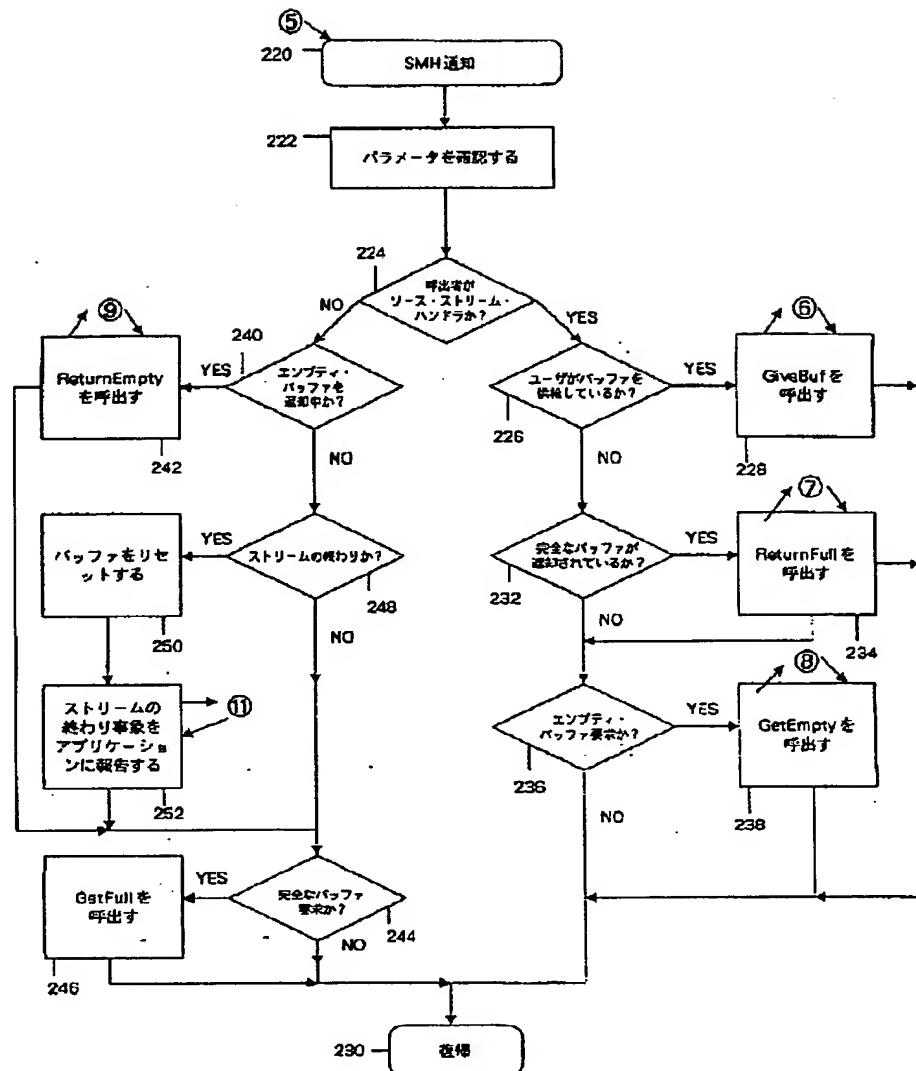
【図7】



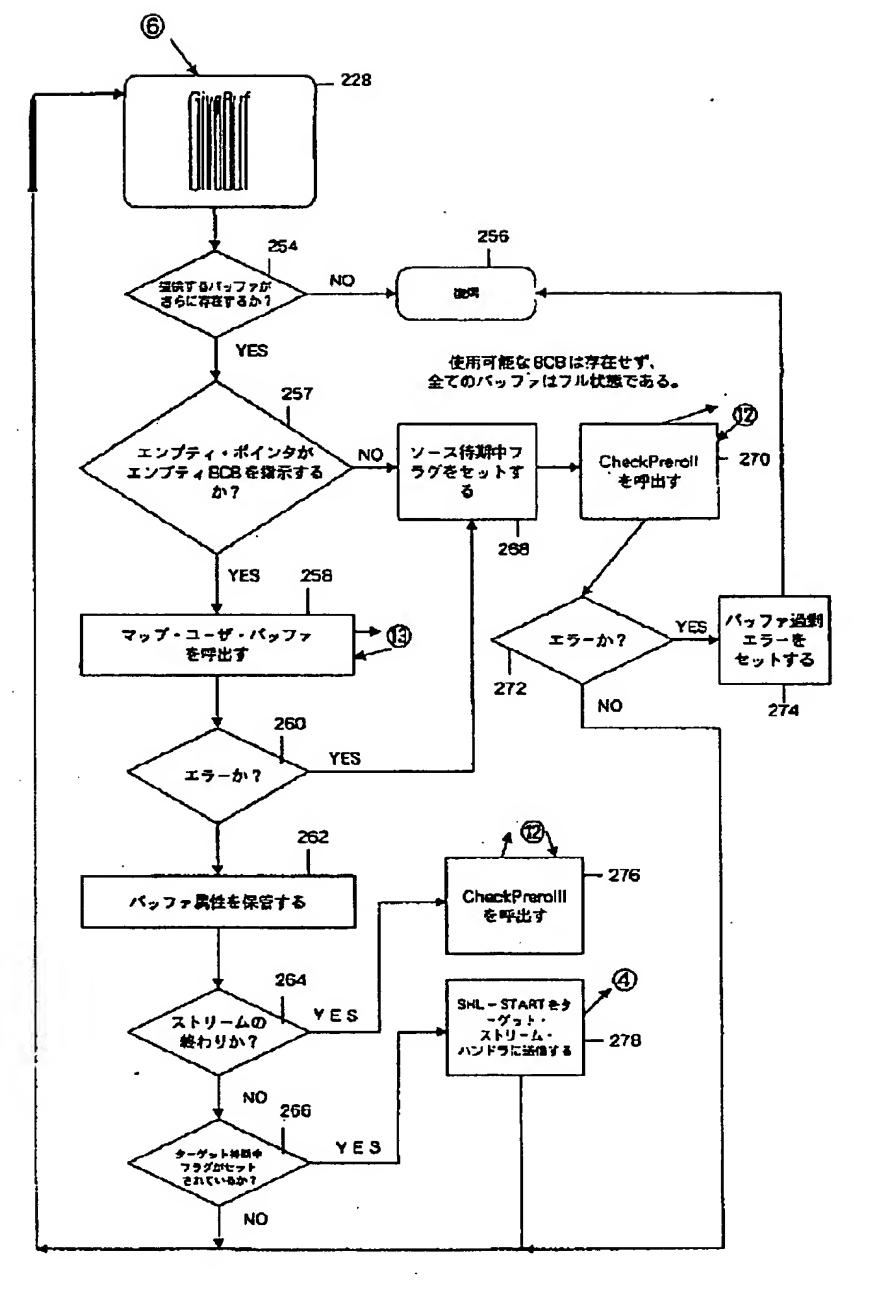
【図8】



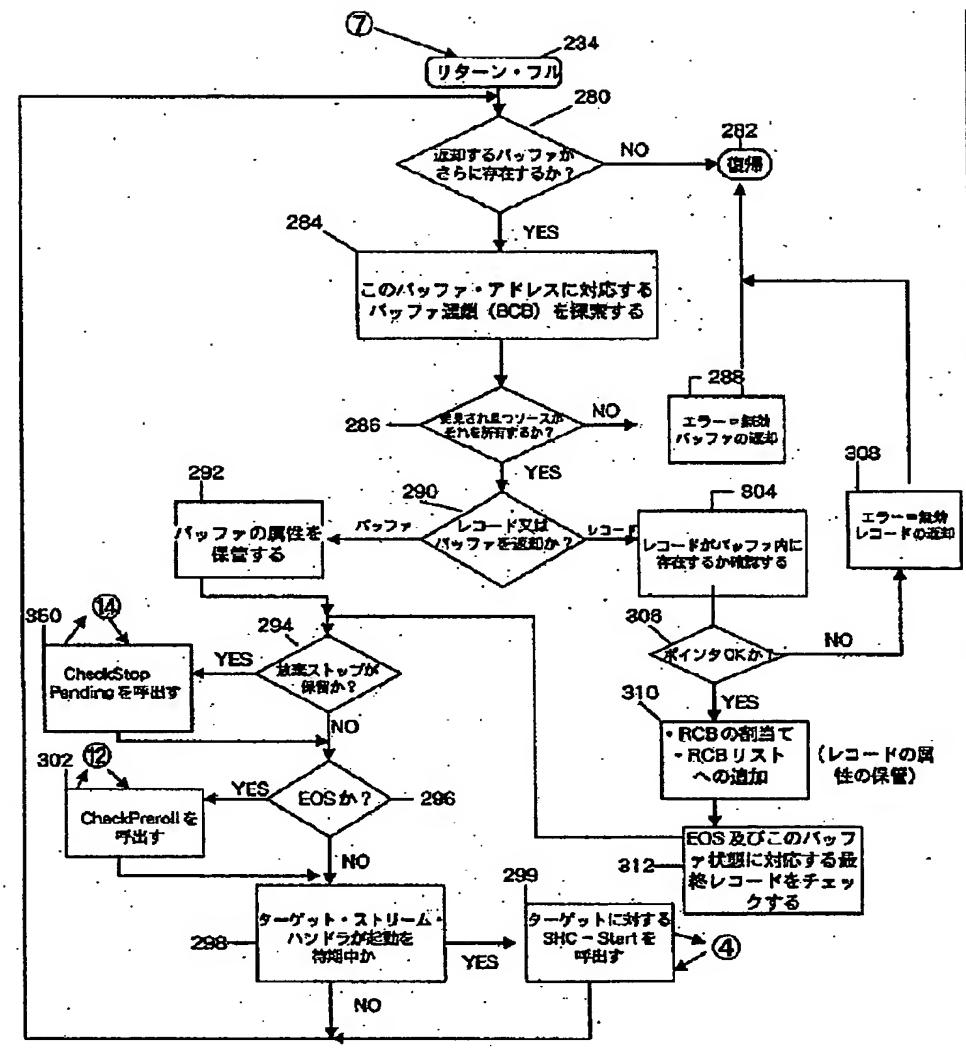
【図9】



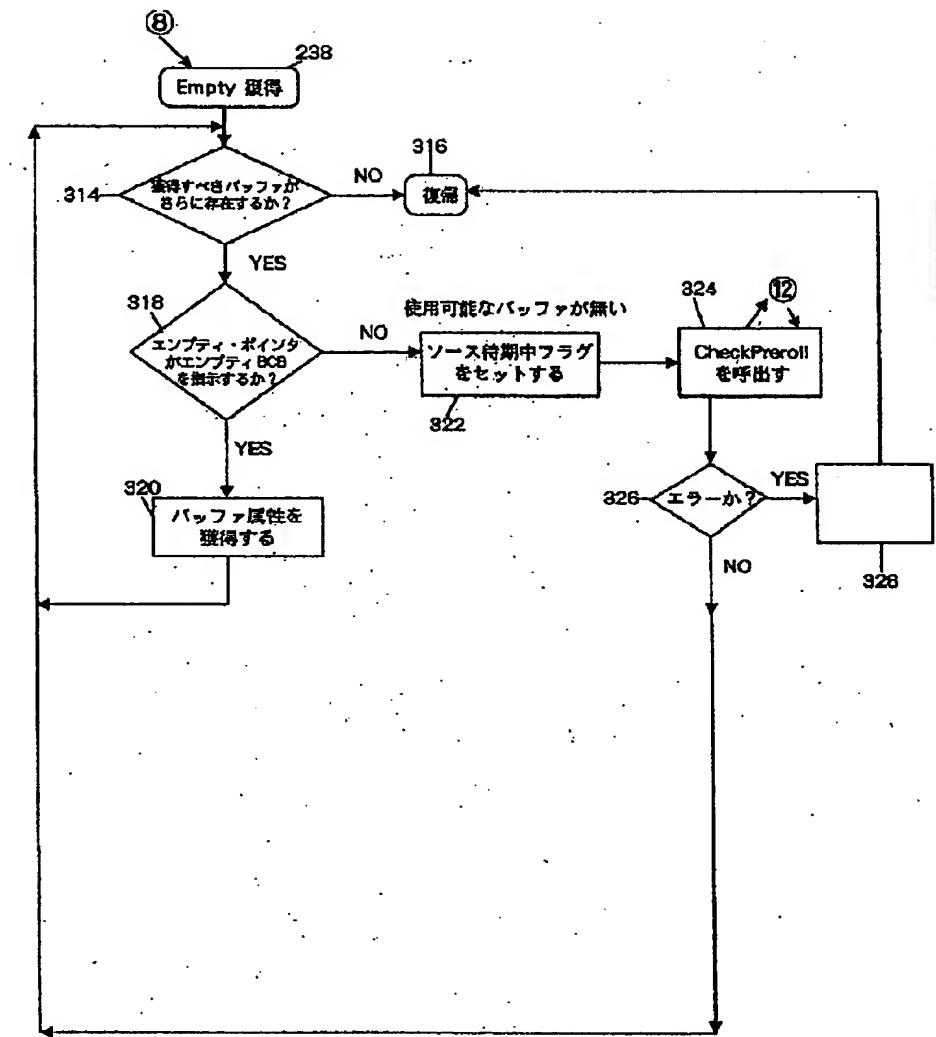
【図10】



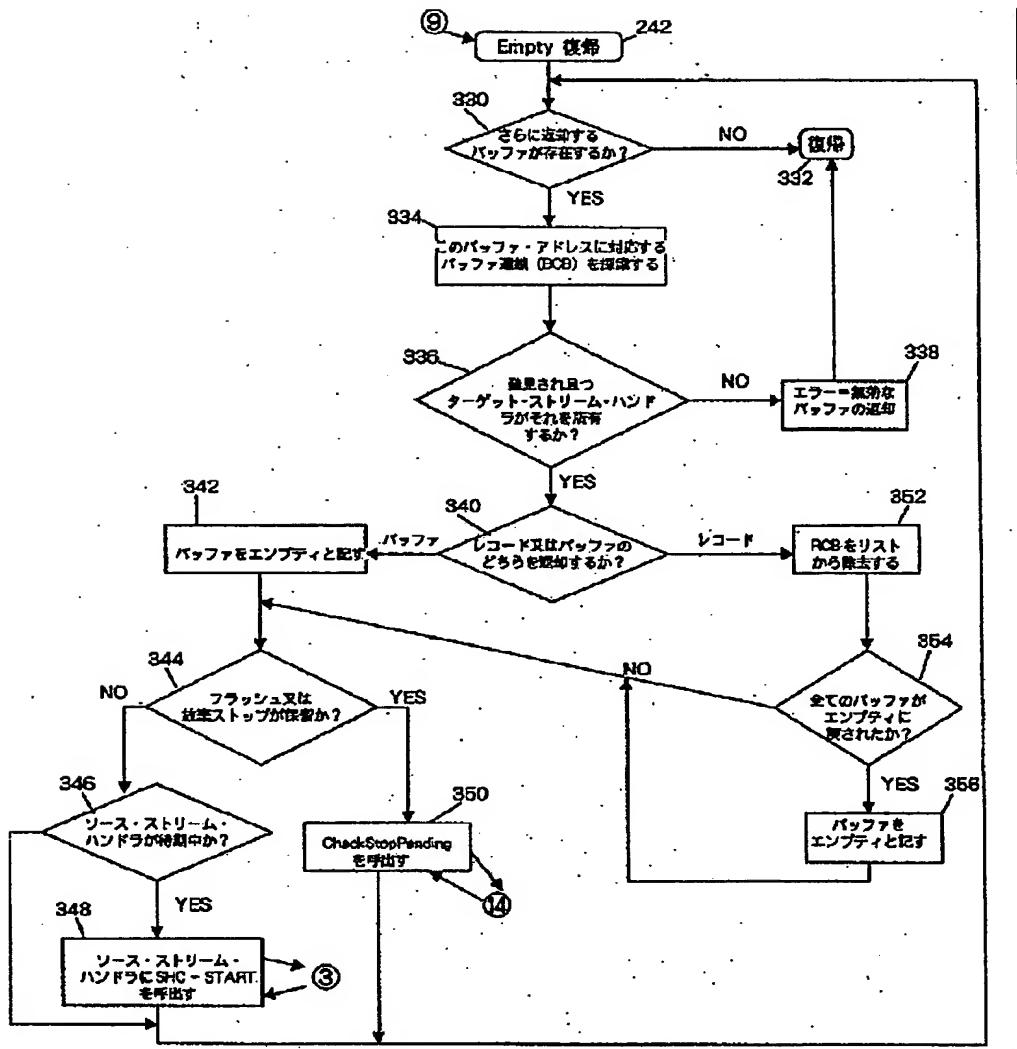
【図11】



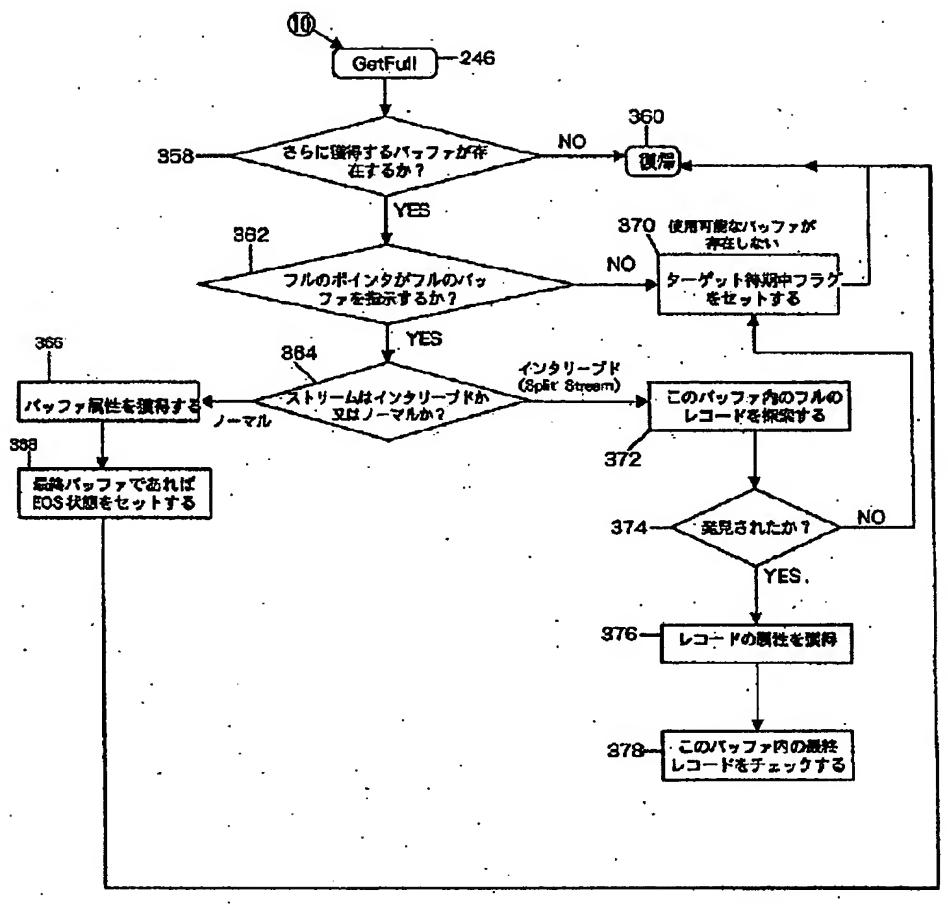
【図12】



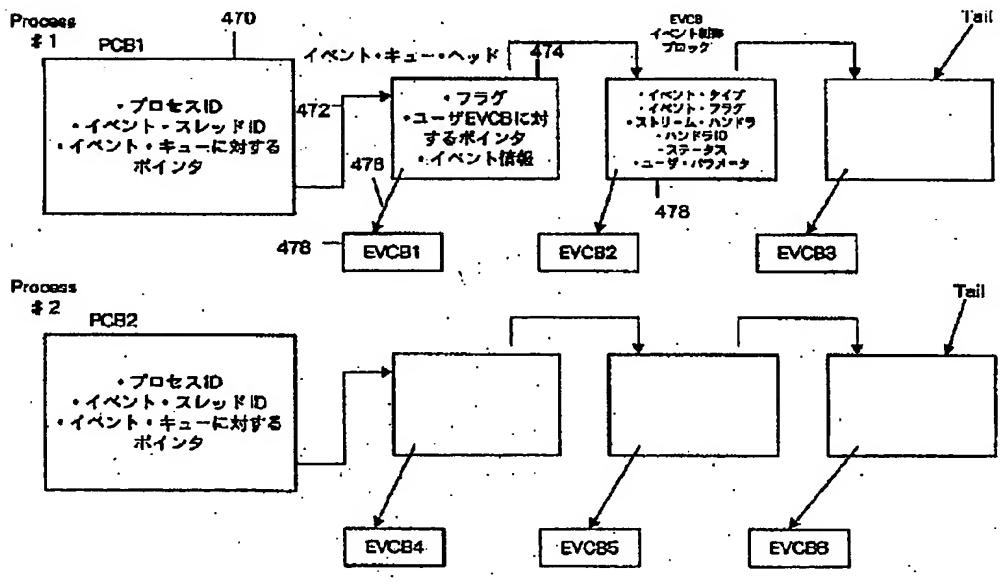
【図13】



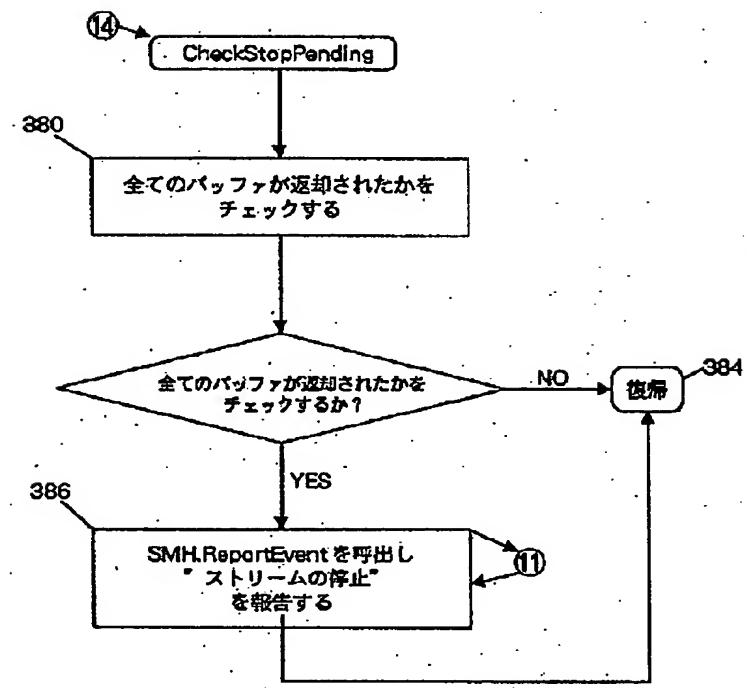
【図14】



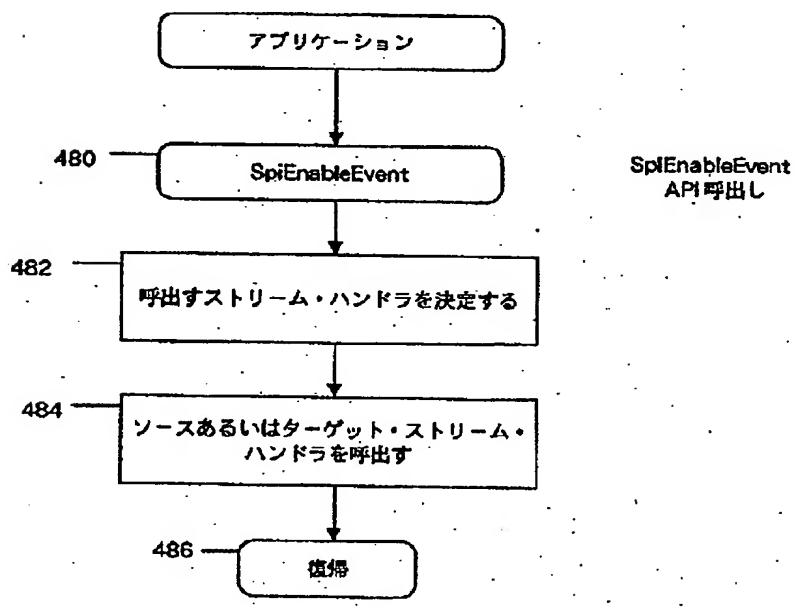
【図22】



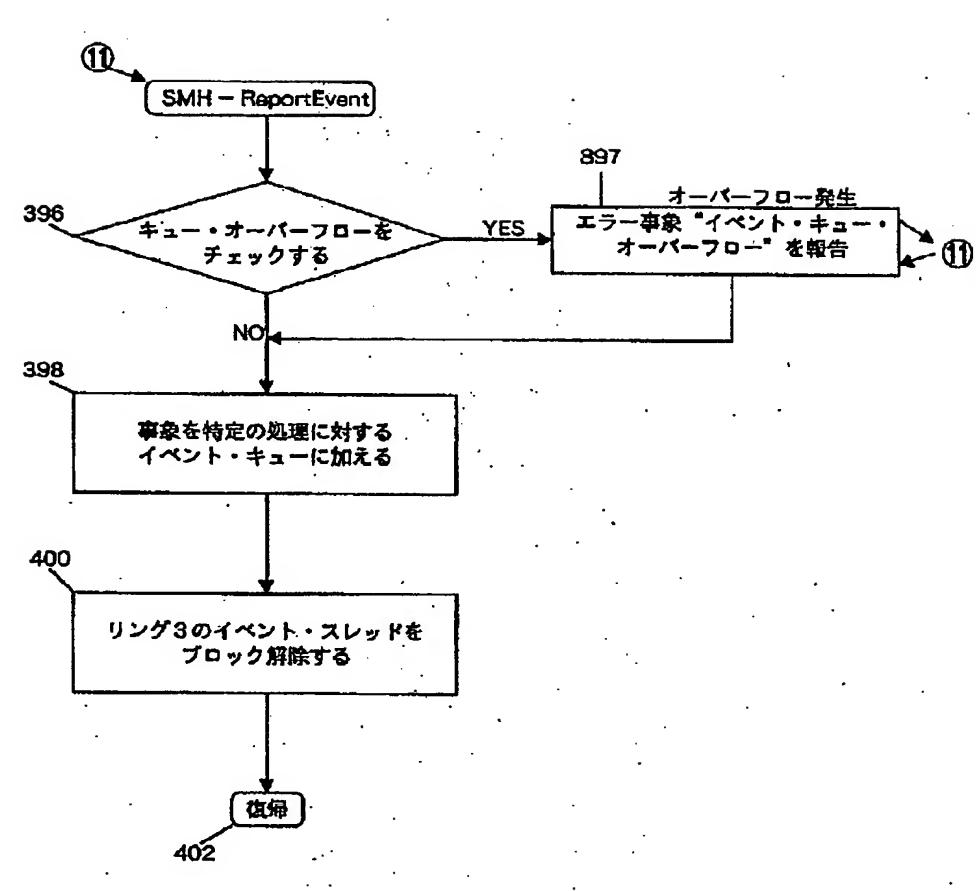
【図15】



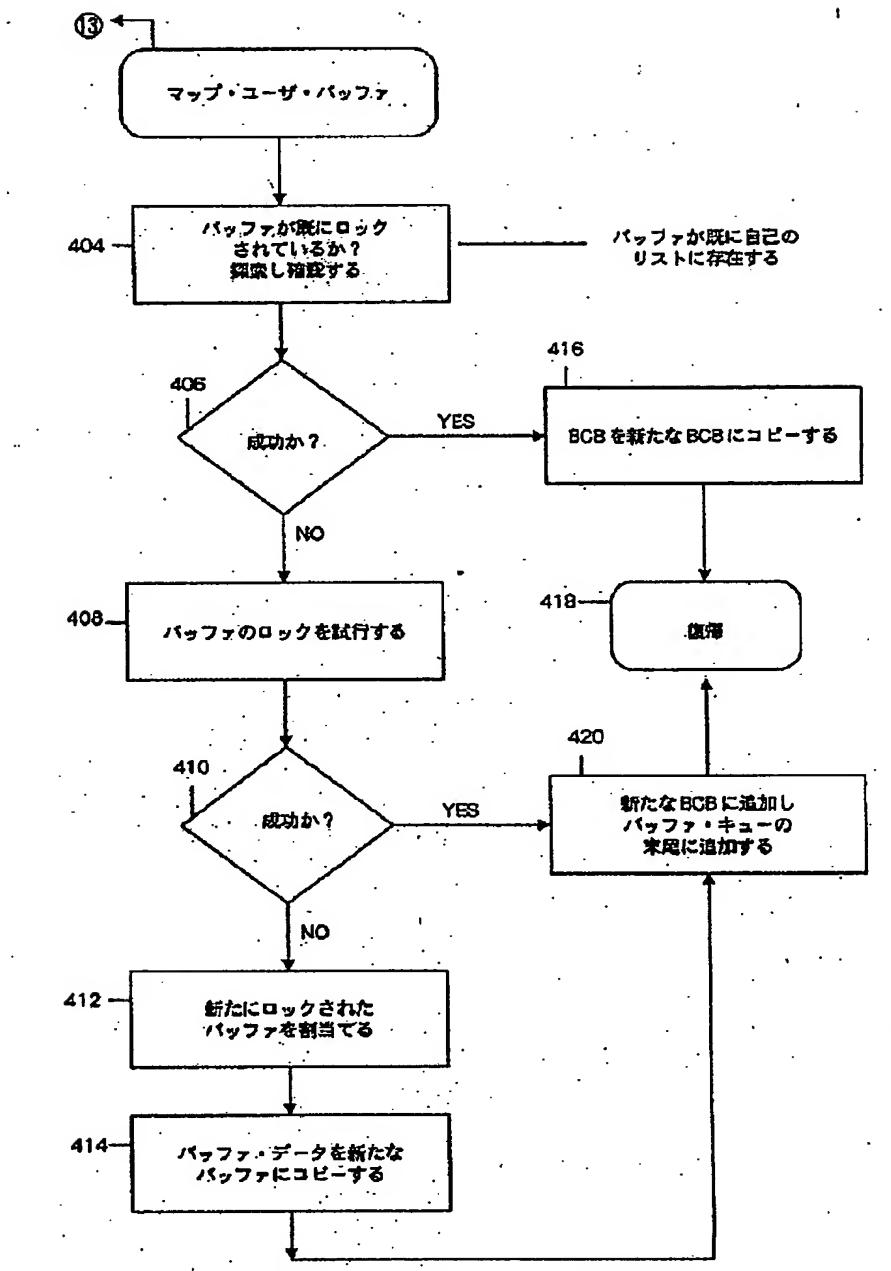
【図23】



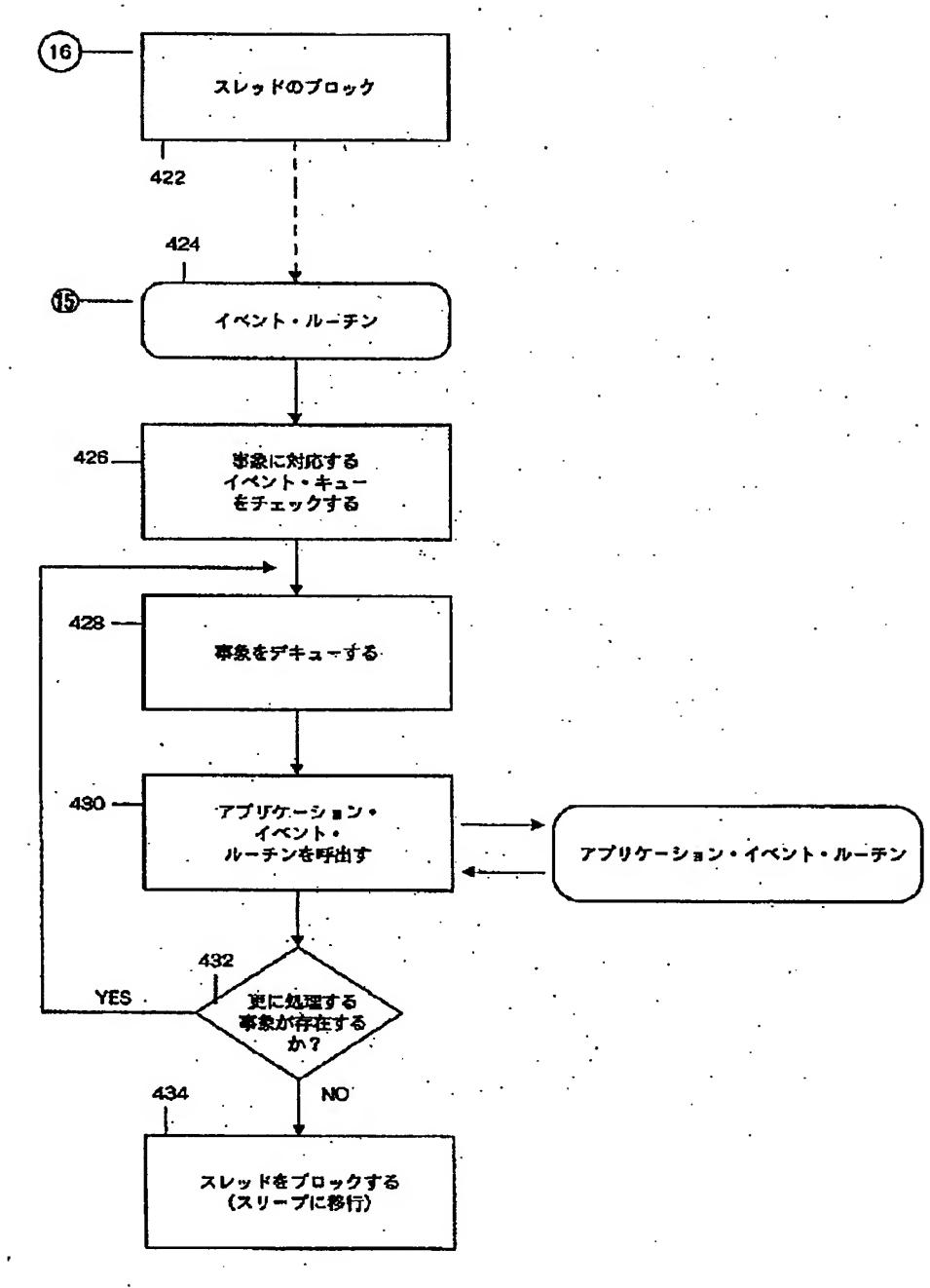
【図17】



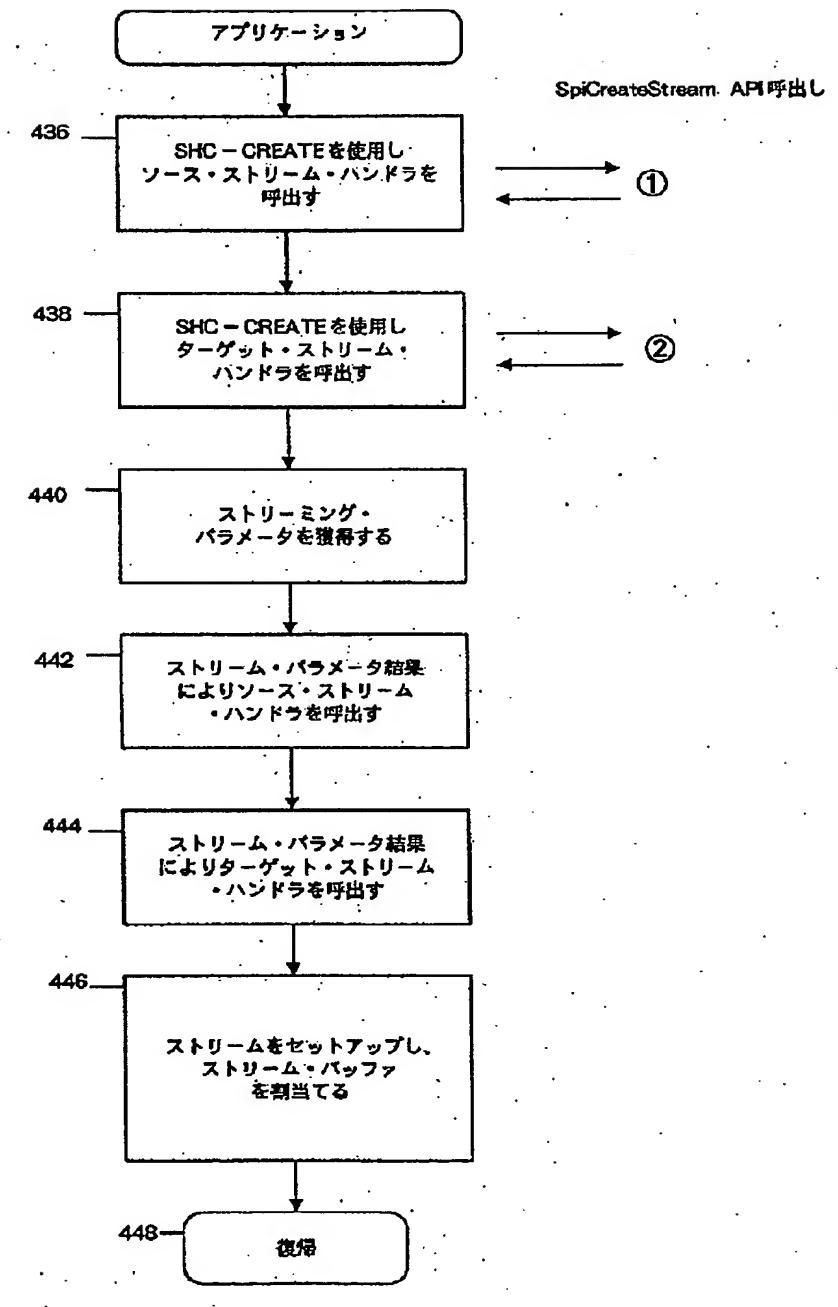
【図18】



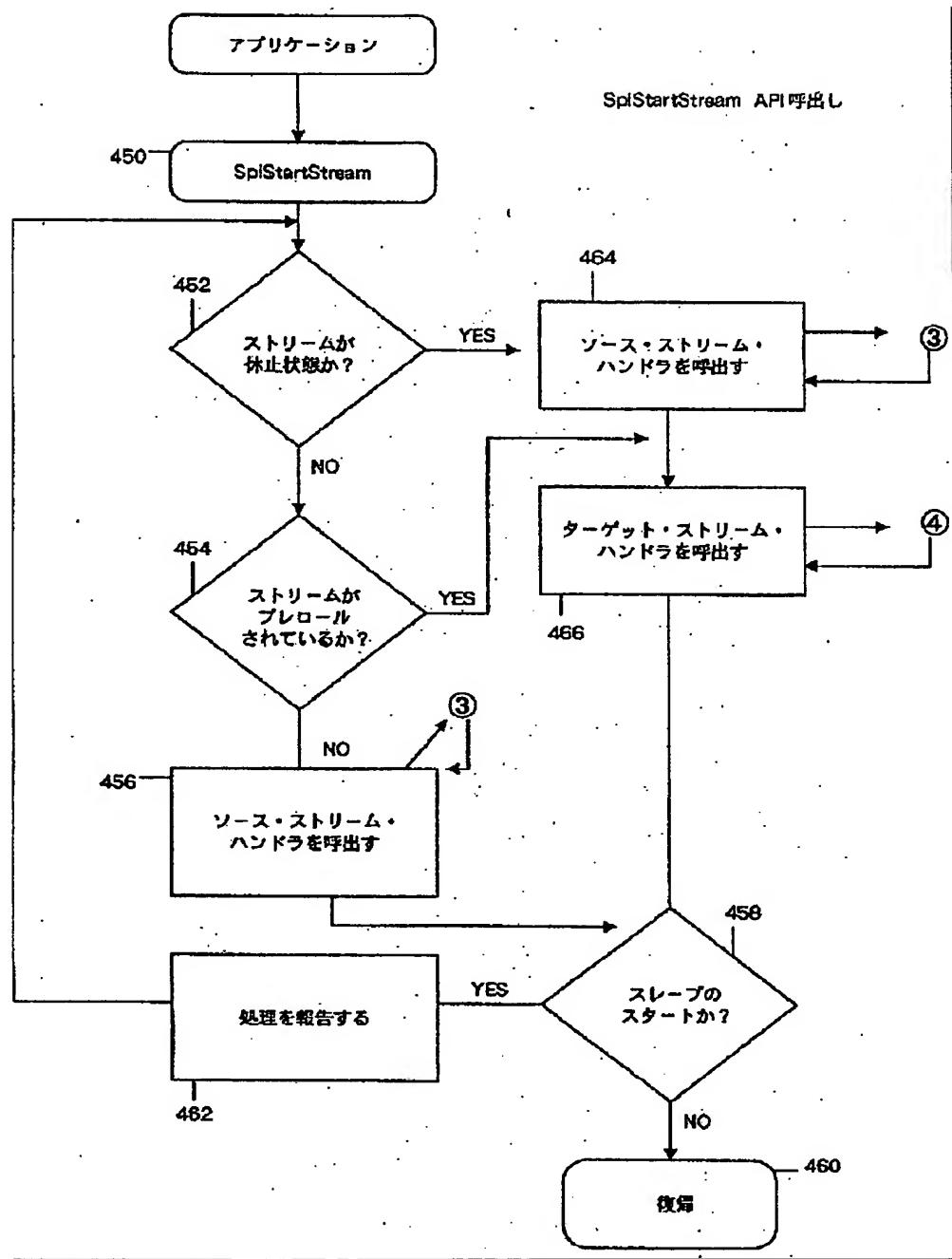
【図19】



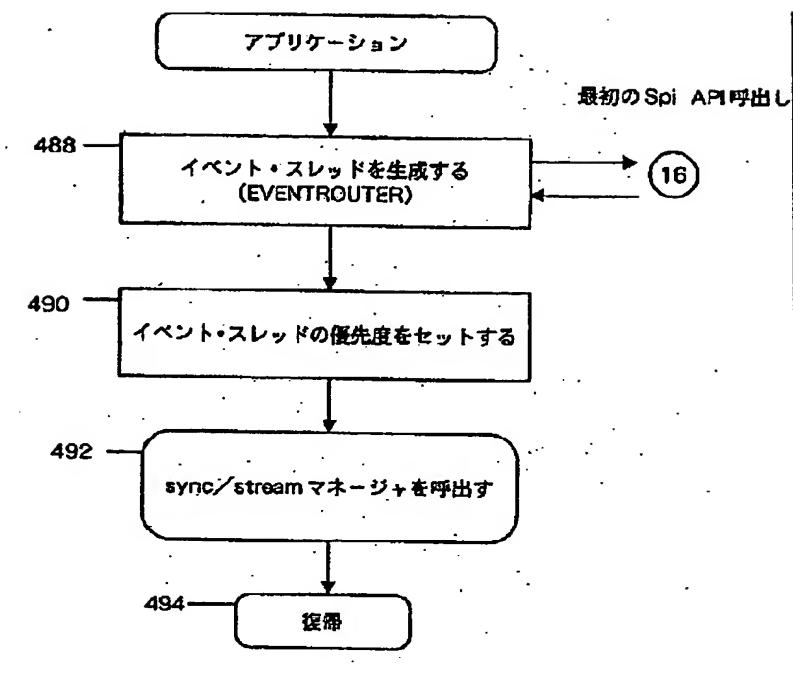
【図20】



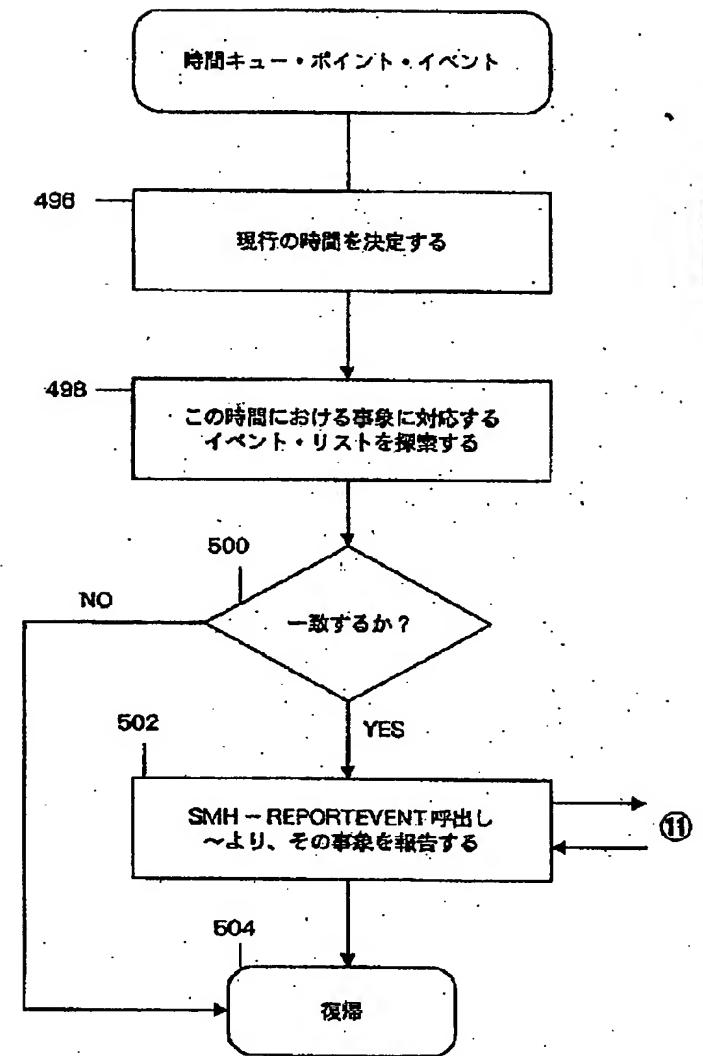
【図21】



【図24】



【図25】



フロントページの続き

(72) 発明者 マイケル・ジェイ・コバル
 アメリカ合衆国33496、フロリダ州ボ
 カ・ラトン、アファームド・レーン
 9080
 (72) 発明者 ウィリアム・ダブリュ・ロートン
 アメリカ合衆国33433、フロリダ州ボ
 カ・ラトン、バターフィールド・レーン
 8290

(72) 発明者 マーティン・ジェイ・ポーラット、ジュー
 ニア
 アメリカ合衆国33444、フロリダ州デル
 レイ・ビーチ、ゼダー・アベニュー
 2409
 (72) 発明者 ジョン・ジー・タイラー
 アメリカ合衆国33437、フロリダ州ボイ
 ントン・ビーチ、ランチバー・ストリー
 ト 10686

(72)発明者 スコット・エル・ウインタース
アメリカ合衆国33324、フロリダ州プラ
ンテーション、ノースウェスト・シック
スス・ストリート 10309

(72)発明者 ゲリー・ジー・オールラン
アメリカ合衆国33431、フロリダ州ボ
カ・ラトン、オーチャード・リジ・レ
ーン 117

(56)参考文献 特開 昭63-186336 (JP, A)
実開 平2-145449 (JP, U)